

digital

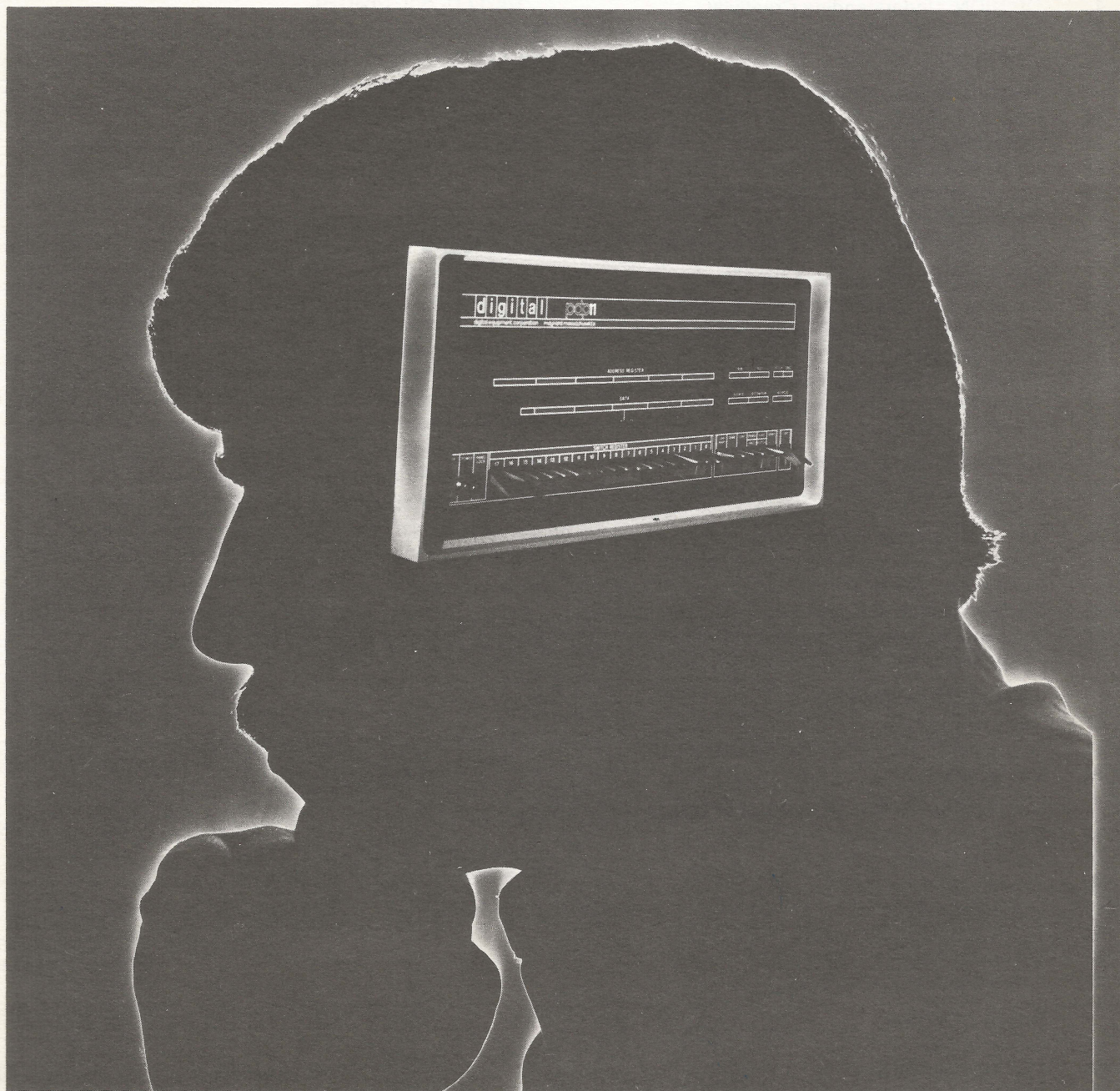
Maynard, Massachusetts 01754

edu

#12

EDU IS FREE

FOR 1974-75...
IF YOU RETURN THE
FORM IN THE
BACK OF THIS
ISSUE



**THIS ISSUE OF EDU FEATURES
COMPUTER SCIENCE AND HUMAN THOUGHT**

THIS ISSUE OF EDU FEATURES COMPUTER SCIENCE AND HUMAN THOUGHT

Page

SPEAK OUT (Opinion and Letters to the Editor)

- 4 OPINION: Understanding Creativity
- 7 OPINION: Computer-Based Learning in 1980: What Will It Be Like?

PRODUCTS AND NEWS

- 10 Newest in Print
- 10 Meetings of Interest
- 12 Introducing EduSystems 100, 200, 250
- 13 Computerized Test Generation
- 14 More About EduTech
- 15 Benevolent Batch

POTPOURRI (Reports and Reviews)

- 16 Teacher Training for Computer Science
- 17 Authorizations, Appropriations, Allocations
- 18 Can Computers Think?
-  20 Computer Science Conference
-  21 About Two Year Colleges
- 21 The Ever-Incomplete Robot

ARTICLES AND PAPERS

-  24 Student-Written Simulations in Laboratory Science
- 25 The Computer and Human Thought
-  26 Computers and Mathematics Instruction—The Third Generation
- 28 On the Science of Computing
-  31 Teaching Assembly Programming with Spacewar

APPLICATION STORIES

- 32 Herkimer County
-  33 A Computer Science Support Laboratory
-  34 Soloworks

SOURCE MATERIAL FOR EDUCATORS

- 38 Computer Science Material
- 39 Book Reviews
- 40 New from Huntington II
-  40 Getting Started?
- 41 New Curriculum Packages
- 42 Computer Tutor
- 42 Cloze Readability Program

IDEAS AT WORK

- 46 Game of Life
- 49 The Mechanical Man
- 50 Kingdom
- 53 Educational Game Reviews
- 54 Program of the Month: Life

PUBLISHER'S STATEMENT

EDU—Issue #12: Summer 1974

Editor: Sally R. Bower, DIGITAL

Contributing Editors:

Robert H. Meese, DIGITAL

David H. Ahl, DIGITAL

EDU is published four times per school year by the Education Products Group, Digital Equipment Corporation, Maynard, Mass. 01754.

Circulation: 15,000

EDU welcomes contributions from its readers. Send to: Editor, EDU, at the above address.

A NEWCOMER

Introducing . . . a newcomer to the pages of EDU—a brand new section entitled “Articles and Papers” that contains a wide range of invited and contributed articles and papers. The “Articles and Papers” classification is intended to bring exceptional reading to the attention of our readership and to provide our authors with the recognition they deserve. Be sure to watch for “Articles and Papers” in future issues of EDU.

. . . And one other small change. To make room for our new arrival, the “Potpourri” section has been reduced to include only reports, excerpts and reviews.

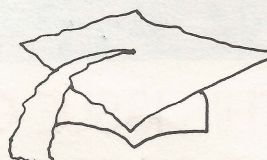
CREDITS-ART AND PHOTO

ART: Bob Barner, 65 Mt. Vernon St., Boston, Mass.
Sally Bower, DIGITAL

PHOTO: Digital Equipment Corporation
University of Pittsburgh

CODED CONTENTS

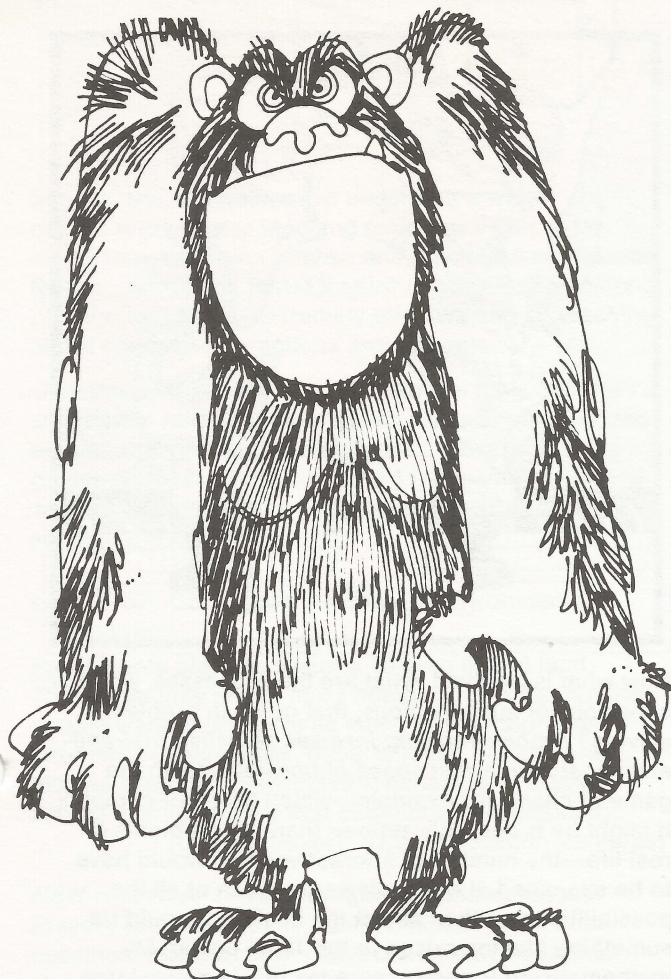
Primary and Secondary
Education =



= Higher Education

SPEAK OUT

(Opinion and Letters
to the Editor)



PERSONALS

Will trade my parallel crosshair cursor for LPS synchronous & asynchronous models. Good condition. Box 109.

I need data to process! You name it, I'll run it. Low rates. Box 43.

Interested in time-sharing with sophisticated systems? Write in complete confidence. Enclose tapes. Box 516. PDP-11 series computer, well-programmed, seeks input/output with late-model hardware. Object, permanent tie-in. Tapes on request. Box 18.

9 1/2-year old computer, some parts damaged, needs repair, looking for not-too-demanding job with company in Miami area. Box 91.

CLASSIFIEDS

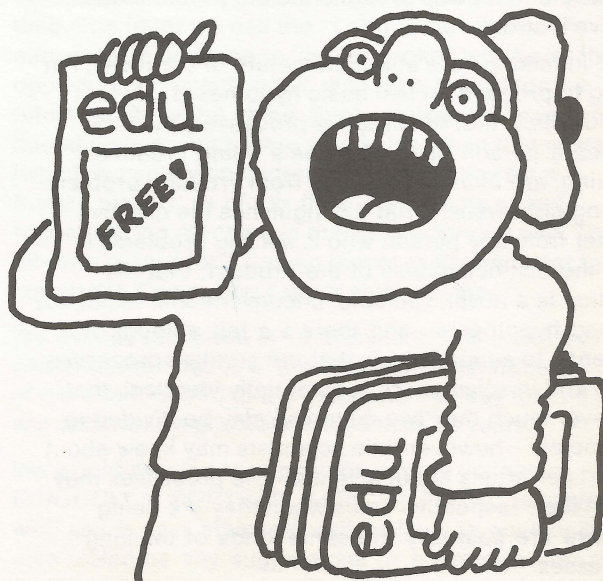
The Association of Teachers of Low Achievers in Mathematics is dedicated to the task of determining more effective means for teaching the low achiever. The association is interested in scheduling meetings and workshops for educators to share ideas and successful methods and programs. For more information regarding the Association contact Mr. Herb Chasan, 56 Agnes Drive, Framingham, Mass. 01701.

EDU IS FREE

We wanted to tell you that Santa filled everyone's subscription for next school year—or that Socrates' fairy godmother left the EDU editor an unexpected inheritance. But we knew that our readers were too smart for that!

So instead, the truth. And the truth of the matter is, that we at EDU understand where that small subscription fee was coming from. So consider the FREE EDU a part of our continually expanding service to the educational community! To receive EDU at no cost next year, simply return the form in the back of this issue.

One additional favor—PLEASE? Keep writing to us—send us your thoughts, your ideas and your stories. Because we still need them—even though EDU is free!



OPINION: UNDERSTANDING CREATIVITY

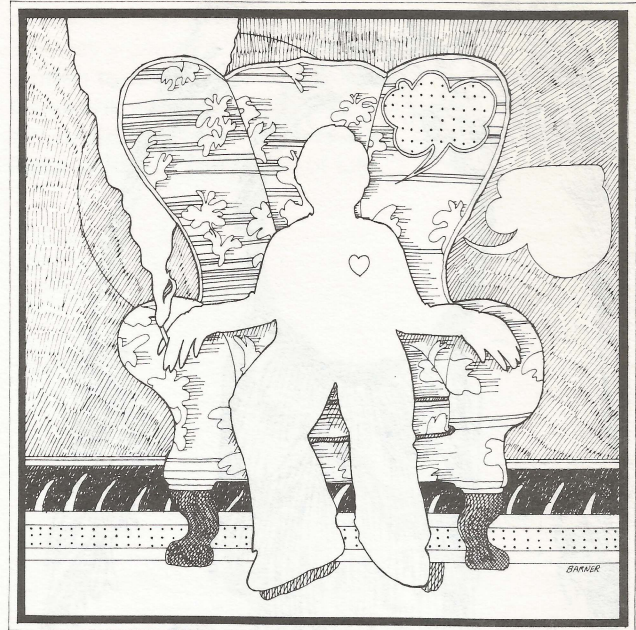
by Herbert A. Simon

You may wonder how the computer comes into the topic of creativity. My first answer is that a computer, though it is a machine, need not behave in a machine-like manner. The term "machine" usually means to us repetitive, mechanical, inflexible, stereotyped behavior—and clearly that isn't what creative behavior is. But by using computers as flexible symbol-manipulators, not simply as devices for doing speedy arithmetic, we have been able to learn a great deal about human problem-solving; and by simulating that problem-solving on the computer we've been able to use computers to learn about creativity.

But the question of how the computer was used to do this, how it has been a tool in psychological research on creativity, is the subject of another article. What I'm going to deal with, ignoring computers for the moment, is what we now know about creativity—about how human beings think creatively.

What do we mean by creativity? When do we call something creative? We call human problem-solving creative to the extent that one or more of the following conditions are satisfied: first, if the product of the thinking has novelty and value either for the thinker or for his society; second, if the thinking is unconventional, in the sense that it requires modification or rejection of previously-accepted ideas. Third, we call thinking creative if it requires high motivation and persistence, taking place either over a considerable span of time or at high intensity. And finally, we tend to call thinking creative if it deals with or solves a problem which, initially, as it was posed, was a vague and ill-defined problem, so that part of the task of the creative thinker was to formulate the problem itself, to give it structure.

Now, in talking here about the nature of creativity, I'm going to proceed on two basic hypotheses. My first hypothesis is that the creative processes, the processes a person uses when he's doing creative thinking, are indistinguishable from ordinary problem-solving processes. What distinguishes the creative thinker from any person who is solving problems is only the distinctiveness of the product: that his solution is a novel, valuable, unconventional result. My second hypothesis—and there's a fair amount of evidence to support it—is that the creative processes in art and in science are substantially identical, that however much the "two cultures" may be divided in our society—however little scientists may know about the arts or artists about science—the processes they use in their respective fields when they are being creative are basically the same kinds of thinking processes.



Now what is involved, what are the processes, whether conscious or subconscious, that go on in problem-solving? *Problem-solving involves selective trial-and-error search in a vast space of possibilities.* In the game of chess, for example—which however complex it might be is certainly simpler than the problems of real life—the number of alternatives that would have to be examined, if a person were to look at all the possibilities and then select the best one, would be something like ten raised to the 120th power. My mathematician friends assure me this is a very large number: indeed, larger than the number of molecules in the universe. Neither human beings nor computers can look at 10^{120} alternatives. Problem-solving does not involve looking at 10^{120} things; it involves looking at the very small number of things that human beings have time to look at in their thought processes. Thus problem-solving is a *highly selective*, trial-and-error search in an enormous geometrically-branching maze of possibilities. This search generates various possibilities for solutions or partial solutions for the problem that has been posed. After they are generated, these possibilities are tested to see whether they are in fact solutions. We have today considerable evidence in human problem-solving that the numbers of possibilities actually examined in this highly selective search is almost always well under a hundred.

In addition to selective trial-and-error search, *problem-solving involves abstraction.* Abstraction in itself is perhaps a rather vague and abstract term, but I mean something pretty definite by it. By abstraction I mean taking the original problem-situation, throwing away a good deal of the detail of that situation, arriving

thereby at a simplified problem, solving the simplified problem (which, because detail has been omitted, is a good deal easier to solve than the original problem), and finally using the solution of the simplified problem, after re-inserting the detail, as a guide or plan for the solution of the original problem. In the studies of problem-solving that have been made, we have seen that this kind of abstraction is one of the very powerful tools that good problem-solvers use effectively. Abstraction aids problem-solving by cutting down the size of that vast maze of possibilities through which the problem-solver must search.

And finally, *problem-solving activity involves searching for and finding patterns*. If I were to recite a string of letters—A, BB, CCC, DDDD—and ask you what the next letters were you would immediately reply, I think, with 5 letter E's, EEEEE. Just by hearing me read this string of letters, you discover a simple pattern, going down the letters of the alphabet, and increasing the number of repetitions of each successive letter. This discovery of pattern is a simple example of behavior that we find universally in problem-solvers; searching in the problem material for various kinds of patterns, and substituting the simple rule that describes the pattern for the complexity of the original problem. It's a kind of behavior that's very prevalent, for example, in listening to and understanding music. The task of the listener is to find patterns in the harmonics and in the melodic line of the tones. His understanding of the music involves hearing these patterns and being able to anticipate the patterns in the notes that are still to come. We find these common kinds of processes or activities—selective search, abstracting, pattern-finding—in all sorts of human problem-solving, whether in the sciences or in the arts.

So much for a very broad description of the problem-solving processes that we see human beings using in general, and the particular forms these processes take when people are trying to be creative. What can we say about the way to be creative? How can we be more creative in our problem-solving activities? What are the conditions under which a person succeeds in producing creative results in his thinking?

I would say that you are likely to discover something that is novel and valuable only if one or more of the following conditions—some of which I have already alluded to—are fulfilled. First of all, you can be smarter than other people—but it's immodest to say that you are and it's usually unwise to *think* that you are. It's much safer to suppose that other intelligent people have tackled the same problems that you're going to tackle; and if you turn up novel answers, which other people haven't thought of, it must be because you are using some secret weapon in your attack on the problem that they didn't possess. It's rather unwise to depend solely on your intelligence as that secret weapon.

But what are the other possibilities, the other things that might allow you to arrive at a unique product? First of all, you may have a new problem. Much of the creativity that is exhibited in our world is exhibited in

relation to new problems that weren't solved earlier because they weren't posed earlier. Today, for example, there are lots of opportunities to be creative about space travel, because space travel is something that people have only recently considered to be realistically possible. It simply was not in the cards a few years ago.

Another possibility that might allow you to be creative is that you're aware of a new phenomenon; you know some new facts about the world that other people don't yet know. You might have come at those facts in a variety of ways. You might have stumbled over them accidentally. There's a nice word to describe that state of affairs: it's called "serendipity." Serendipity consists in having the good fortune to stumble across important facts that other people have not yet seen; however, the accidental observation of a new phenomenon almost always happens to a prepared mind, not to somebody who has only thought about the problem for 15 minutes.

Another way in which a person may discover a novel phenomenon is by using a new instrument. When the electron microscope came along, there were possibilities for observing phenomena at the molecular level that had never been observed before, and thereby, being creative in producing new knowledge with this instrument.

What are some of the other tools or some of the other secret weapons you might apply in order to be creative and arrive at a novel result? One possibility is to invent or acquire a new analytic tool. It was very difficult before the invention of the calculus to think about the movements of the planets and the stars, and the ways in which the forces that one exerted on another might affect these motions. But the calculus was a new analytic tool that enabled Newton and the people who followed him to think through problems that were simply too difficult for the human mind before that time.

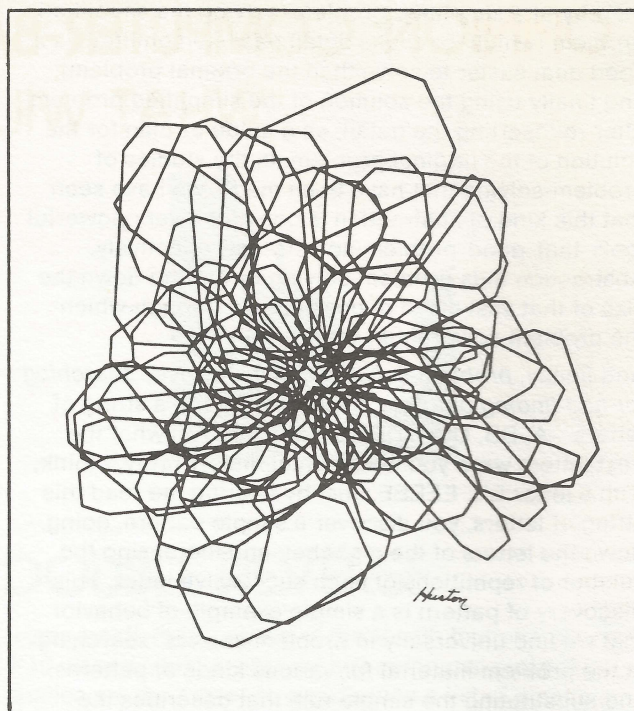


An analytic tool of comparable power has emerged in our own time. The computer program, and languages specially devised for formulating computer programs, give people who use this analytic tool a way of talking about and thinking about complex phenomena that we simply didn't have before the day of the computer. Hence, people who work with these new languages have opportunities to be creative—to solve problems which couldn't be solved before these languages were available.

Another possible source of creativity comes from being able to draw on a mixture of ideas and cues garnered from different fields of knowledge. If you expose yourself to a variety of disciplines or areas of knowledge, you may be able to put together the ideas you get from these different fields in novel combinations and solve problems that others haven't.

Now, the result of being able to apply one or more of these advantages—a superior intelligence, a new problem, some new phenomena that you have observed, a new analytic tool that you possess because you've invented it or gotten it from somebody, or a mixture of cues from different fields—the result of having these techniques available may be that you will remove a barrier to search, you will notice that people have failed to search in the right parts of the problem space, because some kind of tradition has barred out a part of that space; or you will find some new principle of selection, which tells you what part of the space needs exploring, where you should look for the answer to the problem, and as a result you will discover something novel and useful; you will be creative.

Let me summarize the main things that I have been saying here. First of all, I have been arguing that creative problem-solving is not basically very different from any other kind of problem-solving. In considerable part, developing one's creativity is a matter simply of developing one's problem-solving habits and one's motivation. Secondly, I have been saying, or at least my remarks imply clearly, that creativity is not costless. One reason why a lot of people aren't creative, or aren't more creative than they are, is because they are not willing to pay the price. You are unlikely to be highly creative in significant ways unless several conditions are satisfied. First of all, the creativity must be more important to you than lots of other things you might be doing. It must be more important for you to immerse yourself in the problem, be preoccupied with it, than to attend to other matters. And it will not be possible for a person to immerse himself in a problem-situation unless he is able to tolerate ambiguity, unless he feels comfortable in the face of an unstructured and, as yet, unsolved problem, and can maintain that comfort or at least a reasonably low level of discomfort, over a long period of time. Finally, a person is unlikely to be creative unless he has developed the habit of asking, "What is the question I'm trying to answer, and what would a solution look like, if I had one?"



And now it's time, I think, to go back to computers and to the relation between creativity and computers.

Since, in my experience in talking about computers, many people seem to think that these terms are antithetical, almost contradictory, and many people seem to be worried about the impact of the computer on our society—seem worried that the computer in some sense threatens man—I'd like to conclude with a statement of my position on this particular issue. And in doing so I'd like to quote something I have written in another place.

The developing capacity of computers to simulate man, and thus both to serve as his substitute and to provide a theory of human mental functions, will change man's conception of his own identity as a species. The definition of man's uniqueness has always formed the kernel of his cosmological and ethical systems. With Copernicus and Galileo, he ceased to be the species located at the center of the universe, attended by sun and stars. With Darwin, he ceased to be the species specially created and specially endowed by God with soul and reason. With Freud, he ceased to be the species whose behavior was governed by conscious, rational mind. As we begin to produce mechanisms that think and learn, he has ceased to be the species uniquely capable of complex, intelligent manipulation of his environment. I am confident that man will, as he has in the past, find a new way of describing his place in the universe—a way that, however different from the present one, will satisfy his needs for dignity and for purpose.

About the Author: Herbert A. Simon is Associate Dean and Richard King Mellon Professor of Computer Science and Psychology at The Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pa.

OPINION: COMPUTER-BASED LEARNING IN 1980: WHAT WILL IT BE LIKE?

Erik D. McWilliams
National Science Foundation

This article is excerpted with permission from an address presented by Mr. McWilliams at the ACM SIGUCC Symposium on Small College Computer Centers held in conjunction with CCUC/4 in Claremont, California, June 21-22, 1973. Mr. McWilliams directs the Technological Innovation in Education Group at NSF.

Anyone who has experienced, as many of us have, the disparity between promise and performance for computer and other technologies should be rather cautious in predicting future capabilities. However, I was asked by the CCUC Committee to be "provocative" and in that spirit I am prepared to predict that by 1980, Computer-Based Learning (CBL) will be characterized by the availability of a wide range of low-cost, high-performance educational computing systems, capable of providing highly effective instruction for most disciplines and educational levels. Furthermore, CBL will be acknowledged, finally, as the most important educational innovation since printing, and most institutions will either be using it routinely or will be making plans to do so.

I hope that at least some of you consider this to be a "provocative" statement. I believe it to be a reasonable one, however, and one upon which your institution should be informed, especially since, in my opinion, only a severe depression or other national emergency can delay this prediction by more than a few years.

Notice that I've used the euphemism "CBL" instead of "CAI" or other more restrictive term. Within CBL, I embrace all present forms of educational computing, and then some. Our educational system is characterized (fortunately, in my opinion) by diversity, and so it will be with CBL. Such systems will differ markedly in matters of:

- scale (ranging from dedicated, self-contained, single-terminal systems to huge instructional utilities serving thousands of terminals scattered all over the country).
- function (CAT, CMI, CAI, "dual and solo mode" problem solving, simulation, and "browsing").
- facility (libraries of software or courseware programs, programming or authoring services, informational storage and retrieval services, graphical input/output, natural language input/output, and so forth).

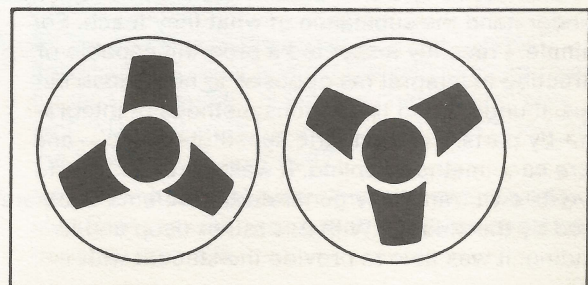
(We shall not want for variety—that is the surest prediction of all!)

Before explaining the basis for this prediction, let me explain the program within the Foundation that I administer, since it provides the perspective for my

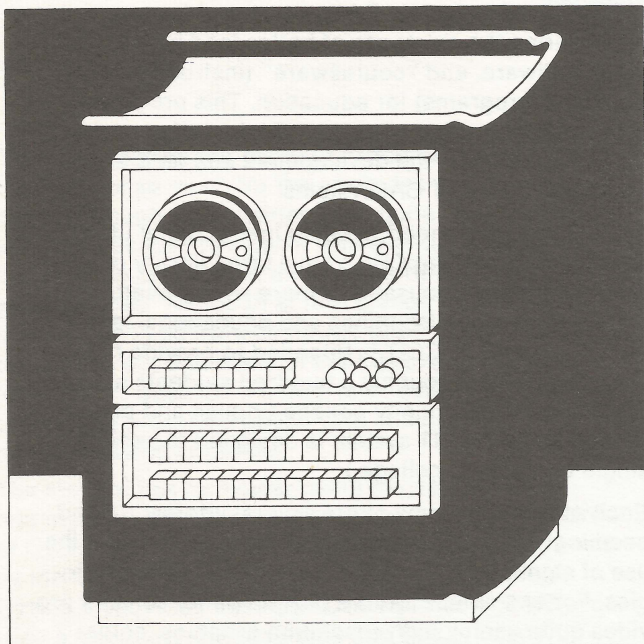
opinions. This program—Technology and Systems—is one of three within NSF concerned with "Technological Innovation in Education," and is responsible for ensuring the existence of better (and better) hardware, software, and "courseware" (instructional computer programs) for education. This program is presently supporting a number of interesting projects, including, notably, the development and field test of the PLATO and TICCIT CAI systems.

PLATO (IV) is intended to be an instructional utility, capable of providing CBL from a large library of courseware to a thousand or more widely scattered students in many disciplines and educational levels, simultaneously. TICCIT is intended to provide highly structured CAI to roughly a hundred students simultaneously, in community college English and mathematics. (One TICCIT system is designed to serve a single community college.)

Each of these systems offers several interesting and seemingly valuable instructional features through the use of significantly different technologies and strategies. For example, PLATO's plasma panel console provides quite useful, high-resolution graphics, color slides, touch input, and audio messages to remote locations using standard telephone lines. TICCIT's "Digicolor" television console will provide high resolution display of color characters or piecewise graphics, plus videotape and audio capability, at a capital cost of less than \$2,000 per console. Furthermore, each system is designed with efficiency as well as facility in mind; each is intended to deliver CAI at a cost—exclusive of the cost for developing the courseware—of \$1.00 or less per student-contact-hour.



The strategies for achieving their educational goals are markedly different as well. PLATO is designed to enable a faculty member to prepare his own courseware, according to whatever educational strategy he prefers. TICCIT's entire design—right down to the "Learner Control Keyboard" and other hardware—is based upon a rather simple, general model of instruction in which the courseware is prepared by teams of "courseware craftsmen," and delivered and executed as data.



I believe that each of these systems will meet its design objectives—namely to provide effective CAI at a cost of \$1.00 per hour—before 1980. By themselves, therefore, they will induce considerable change in the use of CBL, since the cost for human instruction is already several times that figure and increasing steadily. There are many other encouraging developments too, however, and I see strong promise from several that can be grouped loosely together under the heading of "machine intelligence."

For example, computers are already able to communicate with humans in natural (though rather restricted) language, through machine recognition and reproduction of human speech. Natural language communication should be sufficiently far advanced by 1980 as to be quite useful for CBL.

Furthermore, computer programs are increasingly able to understand the substance of what they teach. For example, I recently inspected a program capable of instructing in integral (as opposed to numerical) calculus. It understood the various methods of integration—by parts, trigonometric substitution, etc.—and where each method applied. It was therefore able to solve its own (randomly generated) problems or others posed by the student. With this rather deep understanding, it was able to provide the student with

specific advice (e.g., "perhaps a second substitution of variables would help"). This program has an additional nice wrinkle; it is able to learn from the student. If the student's strategy for a particular problem proves more efficient than the programmed strategy, it is adopted by the program, and will be used for such problems in the future.

There are promising developments in programming languages as well. I'd like to mention one that I find especially exciting.

All programming languages with which I am familiar are sequential, like the machines upon which they execute. In the real world, of course, events proceed in parallel, and often independent of one another. It is not surprising therefore that it has proved difficult to program computer simulations of real world events using standard programming languages, as anybody who has attempted one will attest.

At the same time, simulations of real world phenomena are widely recognized as potentially valuable learning aids. Realistic simulations and graphical displays of ecological and other systems are already popular. The programming required to build (rather than merely use) a simulation prevents many interested faculty members and even some students from attempting to write one. This seems unfortunate, since the insight into the process represented by the simulation will surely be deeper when the simulation is constructed, rather than merely exercised.

One researcher has developed a programming language based upon this notion of parallel processes and other promising concepts, which enables one to program the processes in the manner that he thinks of them. For a space war simulation, for example, one describes the characteristics (speed, acceleration, heading, field of vision and fire, etc.) of one ship, then kicks it off and concentrates upon launching the second ship, or third, or whatever. Each ship proceeds on its programmed course, with co-ordination (the essence of simulation) between ships and earth provided automatically at the systems level. I saw a listing for such a program written in his language; it consisted of a half page of source code!

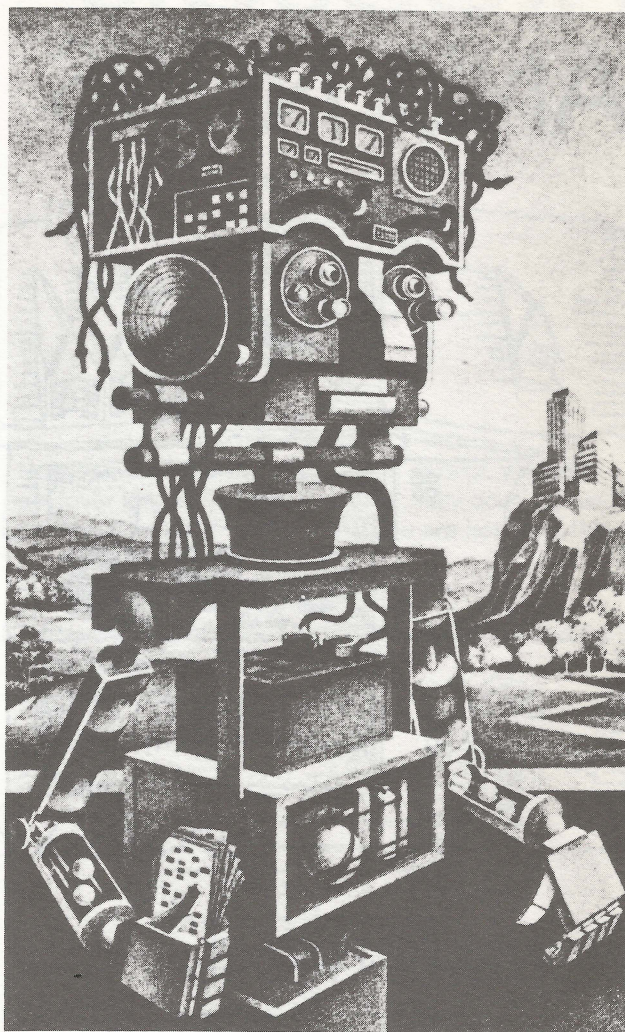
I must mention too the promising attempt to stimulate learning in young children through the use of physical models or other devices, whose principles of operation can be exercised by computer programs written by the children themselves. For example, a computer-controlled, 3-wheeled "turtle" can be moved about by program control (optionally leaving an ink tracing of its circuit around the room) to illustrate the principles of plane geometry. Similarly, the principles of structural biology can be exercised through the use of a computer-controlled physical model of an animal, with moveable parts, "muscles," and "nerve networks."

Computing is also being used to provide services of acknowledged value which most schools have not been able to provide. For example, the Foundation is presently supporting the development and field-test of a computer-based system of career guidance and planning that seems certain to be able to provide college students with more insight into their own values and opportunities, and with more skill in gathering information and exercising choices, than has proved possible heretofore. This system, like others, should be quite advanced and useful long before 1980.

Many other promising technological developments could be cited, but let me mention an equally important phenomenon, namely the concurrent development of the human resources required to exploit this promise. Many people are becoming quite experienced in the application of technology to education, and this experience seems to be growing exponentially. Computing is infiltrating departments formerly considered impregnable. Furthermore, an increasing number of students acquire several years of experience with computing before they leave high school. The interaction between computing-wise students and computing-wise faculty seems to me very promising indeed, especially when supported by theories and conclusions from computing-wise research.

One might conclude from all of this that the millennium is close at hand, except for cost, but here too the signs are quite encouraging. The cost for most basic electronic components has gone down steadily by a factor of 10 in the last 4 or 5 years, and is expected to continue this trend for at least the next 5 years. For example, the cost for computer memory, disk memory, and integrated circuits has been so affected. Concurrently, computer performance and reliability have improved dramatically, through the use of medium and large scale integrated circuits and other technological advances. This has not only improved performance by reducing the distance required for signal travel, but has reduced the physical size of the equipment as well. Computing power that required an entire room 10 years ago could be available in notebook size by 1980, and research is well along in one commercial laboratory to produce just that—a portable, battery-powered computer system, with console, CPU, main and secondary memory, all contained in a notebook. [If this doesn't seem ambitious enough, consider that the head of the project believes that such a "personal computer" could be produced for less than the present cost of a graphics console alone!]

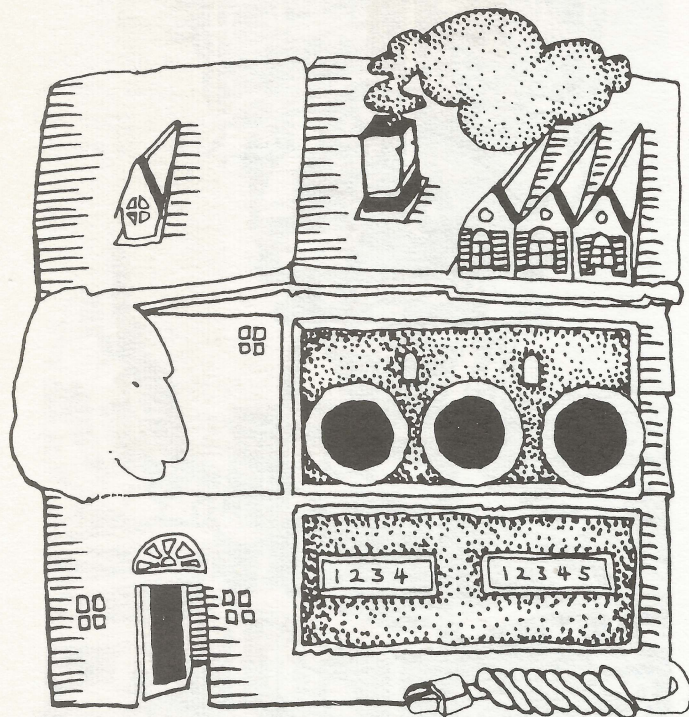
Having thus explained the basis for my prediction, let me hasten to state what I am *not* predicting. I don't think for a minute that even reaching 1980 will be a breeze, or that we will in fact find everything rosy when we get there. Getting access to the technology won't be easy; integrating it into the curriculum will be even harder. A lot of people will be working very hard to do so.



Furthermore, I cannot reassure you that this technology won't displace some faculty. With the cost for human instruction going up and the cost for CBL coming down, some displacement seems inevitable. The way in which this advantage is converted will be a local decision, determined by those local factors, such as growth, that dominate. However, there is little reason to believe that 1980 (or 1984) will be Orwellian (where technology enables institutions to dominate individuals), although we must of course remain constantly on our guard against it.

Some wag recently compared CAI with a waltzing elephant. People, he said, weren't for a minute impressed by the quality of the elephant's waltz, but only that it had the audacity to try! I hope that it is clear from my remarks that I believe such an analogy to be very shortsighted. Within a very few years, that elephant will be doing some very nice steps indeed, and a good many intelligent people will be dancing right along with it. Your institutions, and mine, should be preparing for the occasion.

PRODUCTS AND NEWS



NEWEST IN PRINT

These three new publications are a must for computer users—would-be, growing, and active computer users! Obtain a copy of each by writing to Communication Services, PK1, DIGITAL.

EDUCATION PRODUCT CATALOG

This brochure is a real giant—it discusses all features of the many PDP-8 and PDP-11 educational products as well as selected applications and courseware.

CURRICULUM MATERIAL PRODUCT CATALOG

For the first time in print, a photo-catalog of DIGITAL's curriculum material offerings. Materials are classified into subject areas including mathematics, science, computer science, business, social studies and general studies.

COMPUTERS ARE FOR KIDS

An introduction to the use of computers in secondary schools, this piece contains statistics, testimonials, descriptions of eight instructional and administrative applications as well as product descriptions and DIGITAL support terms. *Computers Are For Kids* is designed to aid faculty members in computer familiarization courses and is also useful for school board presentations and research reports.

MEETINGS OF INTEREST

REGIONAL:

- June 13-15 NCTM, National Council of Teachers of Mathematics, Bozeman Meeting, Bozeman, Montana
- Oct. 17-19 NCTM, National Council of Teachers of Mathematics, Memphis Meeting, Memphis, Tennessee
- Oct. 18-20 NCTM, National Council of Teachers of Mathematics, Minneapolis Meeting, Minneapolis, Minnesota

NATIONAL:

- Apr. 15-19 AERA, American Educational Research Association, Annual Meeting, Chicago, Illinois
- Apr. 17-20 NCTM, National Council of Teachers of Mathematics, Annual Conference, Atlantic City, N.J.
- May 5-8 19th Annual College and University Machine Records Conference, Waco, Texas
- May 6-10 1974 National Computer Conference and Exposition, Chicago, Illinois
- May 7-10 AEDS, Association of Educational Data Systems, Annual Conference, New York City
- May 29-31 1974 Spring DECUS Symposium, Prudential Center, Boston, Massachusetts
- June 24-26 CCUC/5, The Fifth Annual Conference on Computers in the Undergraduate Curriculum, Washington State U., Pullman, Washington

THE 5th ANNUAL CONFERENCE ON COMPUTERS IN THE UNDERGRADUATE CURRICULA

June 24-26, 1974
Washington State University
Pullman, Washington

A meeting of major interest to those involved in computers in higher education. Papers on a wide range of topics will be presented. For information and registration contact:

Dr. Ottis Rechar, Chairman
Computer Science Dept.
Washington State University
Pullman, Washington

It is planned to have a satellite meeting of EDUSIG at CCUC/5. The time and place will be announced at the conference.

1974 SPRING DECUS SYMPOSIUM

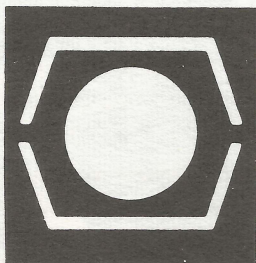
The 1974 Spring DECUS Symposium is scheduled to be held at the Sheraton Hotel, Prudential Center, Boston, from May 29 to 31. In addition to formal papers and sessions, special interest group meetings and DIGITAL product panels will be conducted. In conjunction with the Educational Services Dept., DECUS will present several one or two day seminars and tutorial workshops on subjects of special interest to DIGITAL users. A computer exhibition and literature display will also be available for interested users.

DECUS members from the education community should take special advantage of the wide range of EDUSIG activities. The General Session of EDUSIG will have three main functions; election of officers, discussion of new directions in education, and introduction of independent discussion groups.

Representatives of DIGITAL's Education Products Group will comment on new products for the education community, and an interactive panel session is planned. In addition, educational users will present a variety of papers on topics including "Computer Familiarization," "Logic Games," "Simulation in Social Studies," "The Game of Life—An Open Ended Project," "Computers in Career Education," "Computer Science Curricula in Higher Education," "Applications in the Business School," and "How to Become a DECUS Author."

Demonstrations of administrative and instructional applications, including user-developed programs, are scheduled in the exhibit area, and an informal "drop-in" center will be open to all educational users on Thursday evening of the symposium.

For more information, obtain a copy of the DECUS symposium program from DECUS, PK3-2, DIGITAL.



I cannot conceive that anybody will require multiplications at the rate of 40,000 or even 4,000 per hour; such a revolutionary change as the octonary (octal; a form of binary) scale should not be imposed upon mankind in general for the sake of a few individuals.

—F. H. Wales (1936)



EDUSIG MEETING CALENDAR

DIGITAL's Special Interest Group in Education is having a busy spring! The following EDUSIG meetings have been scheduled:

Mar. 18: NSTA, National Science Teachers Association, Annual Conference, Chicago, Illinois.

The session was held at the Conrad-Hilton from 8-11 p.m. Local users discussed "How To Run A Computer Shop" and computer applications in science. DIGITAL representatives were available for informal discussions.

Apr. 18: NCTM, National Council of Teachers of Mathematics, Annual Conference, Atlantic City, New Jersey.

The session was held at Chalfonte-Haddon Hall from 9-11 p.m. Discussions included the integration of the computer into the school curricula and the successful incorporation of computing into normal teaching schedules.

May 29-31: 1974 Spring DECUS Symposium, Boston, Mass.

EDUSIG plans continuous sessions and papers at this meeting. (See article in this issue of EDU for further details.)

June 24-26: CCUC/5, Conference on Computers in the Undergraduate Curricula, Washington State University, Pullman, Washington.

This meeting is of major interest to those involved in computers in higher education. A special meeting of EDUSIG will be held in conjunction with this conference.

★ INTRODUCING ★

EDUSYSTEM 100

EDUSYSTEM 200

EDUSYSTEM 250

THREE COMPLETELY NEW, LOW-COST, HIGH-PERFORMANCE COMPUTER SYSTEMS SPECIFICALLY FOR EDUCATION

While incessant inflation and serious shortages are wringing educational budgets dry, one company has been working to provide a source of relief. DIGITAL, long the leader in providing low-cost computing facilities aimed at increasing instructional productivity, has now developed the fastest, most powerful, most versatile, multi-user computer systems ever offered in the \$10,000 to \$40,000 price range.

Named EduSystem 100, EduSystem 200 and EduSystem 250, the new systems are based on the 16-bit, PDP-11 mini-computer. They feature extremely fast BASIC language processors which are completely upward compatible for smooth expandability as budgets permit.

EduSystem 100 is the lowest priced of the three new systems at \$10,620 and up. It is a 1-8 user, core only system which uses a very powerful BASIC language processor.

EduSystem 200 is also a 1-8 user system, but it has the added capability of employing files on a mass storage device.

EduSystem 250 adds a "foreground/background" feature to *EduSystem 200* which enables a school or college to run up to 8 users on multi-user BASIC, *simultaneously* with one FORTRAN IV or assembly language job.

Powerful Basic Language

The BASIC languages of the new EduSystems offer most of the features found on DIGITAL's other EduSystems, including:

Calculator mode for immediate response in drill and practice or problem solving situations;

Multiple statements per line for greater programming efficiency and easier program readability;

RANDOMIZE statement for effective simulations of real-world situations;

Dartmouth compatibility for easy transportability of programs among schools.

In addition, a host of new features have been added to all the new EduSystems which means faster, easier, and more efficient programs for students and professors alike. Among the new features are:

Strings of up to 255 characters and 9 string manipulation functions for the ease in text manipulation so essential to precise and understandable output in educational environments;

Program lengths limited only by available core memory, so users need not struggle to complete their jobs under a purely artificial constraint;

PRINT USING statement to enable the format of the output to be specified precisely and easily in the program;

COMMON statement so that data used in one program can be accessed by a second program without retyping;

Assembly language subroutines which allow the computer science student to go beyond the BASIC language, and the professor to add language features specific to his subject area;

And a special for the computernik . . .

Binary and octal conversion functions!



EduSystem 200—one of the three new timesharing systems for educational use.

As if this were not enough, *EduSystem 200* and *250* have even more extras of their own, including, *Overlays and chaining*, enabling not only programs of unlimited length, but also programs which can change themselves under program control;

Direct access files, so that users can access a data bank on a magnetic disk which would be far too large to fit in core memory;

15 simultaneously open files per system, so that different types and sources of data can be used simultaneously.

Expandability

Recognizing that the purchase of a powerful computer system is ideally a gradual process for most schools, DIGITAL has built easy expandability into the new EduSystems, as it has into all its existing EduSystems. Without sacrificing any of the hardware it has purchased with a smaller system and without any changes in the programs written on the smaller systems, a school, junior college or university can make their EduSystem 100 grow to an EduSystem 200 and their EduSystem 200, to an EduSystem 250. The next step is to the most powerful of all BASIC-language timesharing systems—DIGITAL's Resource-Sharing Time-Sharing (RSTS) system.

Flexible System Management

All the new PDP-11 EduSystems feature *variable core partitioning*. With variable core partitioning, the system manager can allocate the user core area among users any way he likes. Thus, he might allow himself enough core memory to run a large class scheduling program, allow three of his users enough core memory to run some relatively large simulation programs, and permit the rest of his users enough memory to learn BASIC by writing short sample programs.

In addition, EduSystems 200 and 250 offer a file protection and user identification scheme. Using the scheme, any user may designate his programs or files as public, write-protected and/or read-protected. Thus, a professor may deny his students access to the inner workings of a simulation while allowing them to use the simulation to *discover* the relationships involved. Or a student may allow only members of his group to use his program while denying access to competing groups in a simulation exercise.

Something for Everyone

EduSystems 100, 200, and 250 are designed to cover the price and capability range for university departments, colleges, junior colleges, community colleges and secondary schools in need of an instructional computing capability.

Typical prices for an EduSystem 100 range from approximately \$10,620 to \$22,000. An EduSystem 200 costs from \$25,250 to \$38,000. Finally, an EduSystem 250 is priced from about \$32,500 to \$39,000.

For more information contact the DIGITAL Educational Sales Engineer in your area.

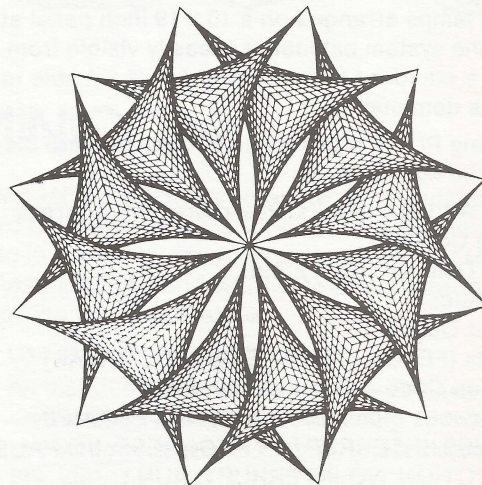
COMPUTERIZED TEST GENERATION

Two new BASIC-PLUS programs are now available from DECUS for PDP-11 RSTS and RSTS/E systems which produce individually tailored tests, quizzes and worksheets. These programs are ideal for generating unique handouts for student review or tests for an entire class.

CEMGEN produces a standard-format ten-question test with different random numeric values substituted in subsequent generations. Consequently, an entire class can be tested on the same objectives, yet students are forced to work independently. CEMGEN also produces an answer key for ease of correction.

The second program, NAMES, when used in conjunction with CEMGEN, creates a file of student names and automatically produces a different test for any number of students. Answer keys may also be produced.

Documentation (no charge) and papertapes containing the computerized test generation programs (\$5) may be obtained from the DECUS Librarian, PK3-2, DIGITAL. Order No. RSTS11-23.



THE PLOTTER AND TSS-8

Plotter-controlling software and monitor patches have been written for EduSystem 50 (TSS-8) by a DIGITAL user in the Department of the Army. The software permits a Calcomp 565 plotter to be interfaced to the PDP-8; plotter operation takes place simultaneously with timesharing.

Additional software is available that allows a BASIC program to indirectly access the plotter. Software and advice regarding the Calcomp 565 and TSS-8 interface may be obtained by writing to Paul Kinzelman, Department of the Army, Harry Diamond Laboratories, Washington, D.C. 20438.

MORE ABOUT EDUTECH

You probably read about our exciting new EduTech system in EDU #9. EduTech, as you remember, is designed expressly for career instruction in computer science/technology and computer maintenance. It consists of the following:

1. A standard PDP-8 based EduSystem of the kind you know and love, namely EduSystems 10, 15, 20, 25, 50 or OS/8, depending on your preference and pocketbook.
2. Some snazzy hardware features (one included, one optional) which tailor the system for computer technology instruction.
3. A complete curriculum package which provides the computer technology instructor with the resources he needs to utilize the system effectively.

In this issue, we'd like to provide some more detailed information on Items 2 and 3, which together make EduTech the unique instructional offering it is.

Special Hardware Features

Included with each EduTech system, and shown in the accompanying photo, is a panel which displays the major active registers and status (MARS) of the PDP-8 processor. The *MARS display* (KC8-HA) consists of 84 1/2-inch lamps arranged on a 10 x 19 inch panel at the top of the system cabinet; it is easily visible from the rear of a standard classroom and thus suitable for whole-class demonstration.

The following PDP-8 registers and conditions are displayed by the MARS panel:

- Memory Address and Extended Memory Address
- Program Counter
- Memory Data and Data Field
- Accumulator
- Multiplier Quotient
- State Bits (FETCH, DEFER, EXECUTE, BREAK)
- Instruction Code and Instruction Field
- Miscellaneous Indicators (LINK, ION, BREAK IN PROGRESS, INTERRUPT IN PROGRESS, I/O PAUSE, MD DIRECTION, NO INTERRUPT, RUN.)

The panel displays the current values of these registers and indicators whenever the rotary switch on the front panel of the PDP-8/E is in either the OFF, MD, or STATE position.

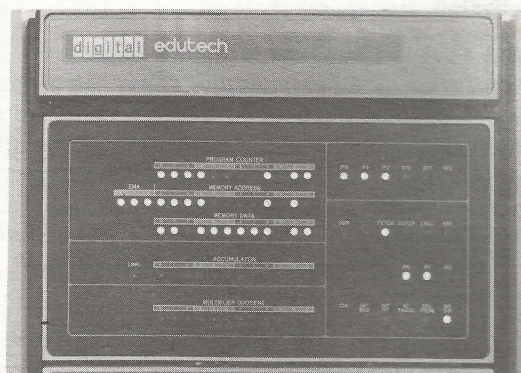
An optional feature available for EduTech Systems is the *Extended Programmer's Console* (KC8-HB).

The console consists of a 5 1/4 x 19 inch panel containing 20 switches, a BNC connector, and a rotary potentiometer knob. The switches represent a 15-bit memory address and five function switches. The console is capable of halting the PDP-8 processor anytime the memory address currently referenced by the processor matches the address set in the KC8-HB console switches by the user. Depending on the setting of the function switches, the pro-

cessor will halt on a FETCH-cycle address match, a data-break address match, or a match occurring on any cycle other than FETCH.

Following the halt, the processor can either be re-started manually or under automatic control of the KC8-HB console. In automatic mode, the processor will execute each instruction following the address match stop after a time interval determined by the setting of the potentiometer knob. The time interval may be varied from 70 milliseconds up to about 9 seconds. It is thus possible for the instructor to explain a series of computer instructions as *they unfold automatically* at a rate that is comfortable for him.

Using the function switches, it is also possible for the console to inhibit the processor from halting, but instead to send a pulse to the BNC connector each time an address match occurs. The pulse output is TTL compatible and may be used, for example, with an oscilloscope.



Used together, the MARS panel and Extended Programmer's Console form a powerful tool for illuminating the internal operation of the PDP-8/E in the computer technology classroom. More broadly, these options should be of interest to any PDP-8/E owner who spends much time in software development/debugging or hardware debugging. By enabling the user to trace the course of his program visually, to isolate and retrace "errors that should never occur," these options will shorten program-development time significantly.

Curriculum Package

No instructional system is really complete without a kit of classroom-tested curriculum material which ensures that the teacher can make effective use of the system from the day of installation. Much of the material included with EduTech has been proven in hundreds of classroom hours in DIGITAL's own training courses for Field Service Engineers. The primary component is the Teacher's Guide and a set of keyed overhead projector transparencies; the outline of this Guide was presented in EDU #9. The remaining components of the EduTech Curriculum Package, which constitute additional instructional resource material,

are as follows:

Small Computer Handbook: DIGITAL

Introduction to Programming: DIGITAL

EduSystem Handbook: DIGITAL

Logic Handbook: DIGITAL

Logic System Design Handbook: DIGITAL

Computer Lab Teacher's Guide: DIGITAL

PDP-8 Programming by Robert W. Southern: Algonquin College Press, 1971.

PDP-8/E Maintenance Manual, Vol. 1: DIGITAL

Digital Computer Fundamentals by Thomas C. Barte: McGraw-Hill, 1972.

Experiments in Logic Design and Computer Interfacing: DIGITAL

For more information on EduTech, order the new EduTech brochure from Communication Services, PK1, DIGITAL.

CARD READER HANDLER

The EduSystem 20 and 25 Card Reader Handler is a patch to EduSystem 20 and 25 software that permits batch jobs, prepared on EduSystem 30 mark sense cards, to be run from any terminal or to a line printer simultaneously with timesharing. In most cases, batch programs originally written for EduSystem 30 will require no modification. The handler requires the latest versions of EduSystem 20 and 25 (DEC-08-ED20B-A and DEC-S8-ED25A-B). EduSystem 40 and 45 users should take note of this offering, too!

The handler and associated documentation are available without charge from the author, Bob Stolarz, 66 Homer Avenue, Apt. 102, Cambridge, Mass. 02138. Please specify either EduSystem 20 or 25.

BENEVOLENT BATCH

DIGITAL's newest mark sense batch system makes money manageable and learners lively

Can a classroom computer pay its way?

You bet it can! And here's one solution that proves it. OS/8 MARK SENSE BATCH is DIGITAL's newest educational batch system, and it's designed to demonstrate that low per-student cost does not have to mean low per-student learning.

The keys to the system's performance are two completely new human-engineered mark sense cards, one for the BASIC language and one for FORTRAN. Both cards use a marking system which is easy to learn and efficient to use, eliminating hours of tedious toil with a template and the missing marks caused by confusing codes.

The system features two high-level languages, (BASIC and FORTRAN IV) which are interchangeable in a single batch stream. The availability of the two languages means that a computer science instructor can help his students discover the similarities and differences of two of the most popular languages, and that he can illustrate how easy it is to learn a second language once a student knows the first one. Since FORTRAN and BASIC programs can be intermingled, students can learn and experiment with the languages at individual rates. Both learners and lecturers can choose the language they find most suitable for any given task.

The low cost per student of OS/8 MARK SENSE BATCH arises from the almost unbelievable "throughput" of the system. A school or college using the system only during normal school hours, nine months per year, could teach with and about computers for less than a nickel per program! Evening and summer utilization cuts the cost even lower.

For more information on OS/8 MARK SENSE BATCH, call your DIGITAL Educational Sales Engineer.

OS-8 Batch BASIC mark sense card.

POTPOURRI (Reports and Reviews)



EDUSIG NEEDS YOUR HELP

We need nominations and volunteers for the EDUSIG officers for next year!

Nominations:

- Chairman
- Associate Chairman for Secondary Schools
- Associate Chairman for Junior and Community Colleges
- Associate Chairman for Higher Education

Volunteers:

- Newsletter for Primary and Secondary Education*
 - Contributing Editors
- Newsletter for Higher Education
 - Contributing Editors
- Meetings Chairmen

Chase Ambler, of the Asheville (NC) School, publishes *Eduhelp*, a secondary school user newsletter. Chase needs editing help and article contributions if *Eduhelp* is to become an effective newsletter for you!

Send your nominations and volunteer offers to EDUSIG Nominations, Education Products Group, Bldg. 5-5, DIGITAL.

TEACHER TRAINING FOR COMPUTER SCIENCE

Project PRISE (Pennsylvania Regional Instructional System For Education) is a three-year NSF-supported project that began in 1971. The project was designed to develop, implement and evaluate prototype teacher education programs in educational uses of computer technology.

Project PRISE was organized into three loosely structured centers: University of Pittsburgh (Western), Pennsylvania State University (Central) and Lehigh University (Eastern). A paper, written by Fredrich H. Bell of University of Pittsburgh, entitled "Computer Science Education Programs For College and Secondary School Teachers" outlines the background and the following conclusions reached in the Western Region program.

The center for the western region activity was the University of Pittsburgh, supported by a DECsystem-10 timesharing system located at the University's computer center. Five western colleges participated including Villa Maria, Gannon, Mercyhurst, Westminster and Slippery Rock Colleges.

Participants attended eight weeks of classes over a one-year period; classes included topics such as BASIC programming, curriculum development, hardware and software fundamentals. In addition, attendees were instructed on a wide range of educational computer uses including classroom administration, instructional management, problem solving, simulation, student-controlled computing and inquiry-discovery learning.

Several important findings were made as a result of the project:

- (1) Hands-on learner-controlled computing is the most effective mode of computer use in education, where effectiveness is evaluated in terms of student interest and significant achievements.
- (2) Computing and computer science should eventually be taught as an applied discipline, giving emphasis to real-world applications.
- (3) Training personnel should have experience in teaching, curriculum development, computer hardware and software.

This first project motivated a second experiment for developing and testing a training program for doctoral students, student teachers and in-service secondary teachers. At this time two courses (one graduate, one undergraduate) have been developed at the University of Pittsburgh using the DECsystem-10 timesharing system.

THE GRANTS DATA BANK

The newest and most efficient instrument to match the interests of those seeking funds with those granting funds is the Grants Data Bank.

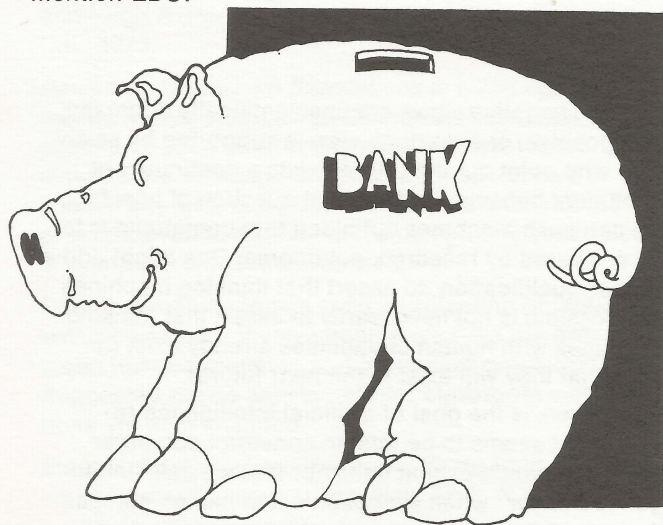
Take, for example, a grant to bring more bluebirds back to New York. The words "bird life" are fed into the bank, and after 90 seconds of polite whirring the computer prints out five grants. (With electronic thoroughness, the computer also prints out a grant to Blue Bird Circle in Houston to construct a new outlet store.) A name such as "Audubon" would undoubtedly have turned up more possibilities!

The price for a search (which can be arranged by mail) is \$15 for up to 50 grant records and 20¢ for each grant record thereafter. For a broad field such as health, education or conservation, specify the maximum expenditure and narrow the field as much as possible. For example, specify only those 15 foundations that have the largest grant programs for a given field.

For more information contact:

The Foundation Center
888 Seventh Avenue
New York, N.Y. 10019

Mention EDU!



Teachers, in general, recognize the oft repeated need for an increased amount of time which can be used to provide individual attention for each student. While utilizing an instructional system, the teacher must become a manager of instruction who is providing guidance, materials and individual interaction. Then the teacher is no longer the dispenser of information and repeater of minute skill details. So long as the hardware and software provide the 'stuff' of the curriculum, the teacher is able, without conflict, to perform that uniquely human role which only he can perform—relating to the needs and potential of each student.

—Maureen Sullivan
in *Instructional Technology*

AUTHORIZATIONS, APPROPRIATIONS, ALLOCATIONS

Remember July 1, 1973? A new Federal fiscal year started. And do you remember Congress authorizing all that money for education? Six billion dollars for OE alone. Have you wondered where it is?

Well, civic fans, to know where it is one must understand the difference between an "authorization" and "appropriation," two important ornaments of the legislative vocabulary. To recipients of Federal funds, the distinction between "authorization" and "appropriation" is the difference between the ability to type and writing a best selling novel, or between swinging a golf club and making a hole in one.

When members of a Congressional committee submit a bill for a vote by the Congress as a whole—the Elementary and Secondary Education Act, for example—they customarily include language "authorizing" the expenditure of certain amounts of money to carry out its provisions. In doing so, however, they have only voiced a sentiment, not gone to the jeweler to buy a wedding ring. The decision to make that kind of commitment originates with the Senate and House Appropriations committees. Working from the President's proposed budget, they draw up and submit to their respective bodies a separate, special bill specifying the actual sums that may be spent for these programs during the next fiscal year. In short, authorization says, "Could be"; appropriation says, "Here we go!"

In any event, the OE appropriations bill was finally passed and signed in February 1974, eight months after the start of the fiscal year. OE funding is \$5.9 billion for FY '74, with another \$75 million for NIE. The biggest allocations were \$1.45 billion for local school district programs for disadvantaged children and \$494 million for vocational programs in public schools.

There are no funds specifically allocated for computer-oriented programs or even educational technology. Nevertheless, there are at least 10 assistance programs under which computers may be purchased or educational computer programs executed.

For more information, you may obtain free a single copy of the "Guide to OE-Administered Programs, Fiscal Year 1974" from:

American Education
P.O. Box 9000
Alexandria, Va. 22304

CAN COMPUTERS THINK?

"A Bird is an instrument working according to mathematical law, which instrument it is within the capacity of man to reproduce with all its movements."

Leonardo da Vinci (1452-1519)

Intelligent Machines and Today's Digital Computer

A common attitude toward today's computers is that such machines are strictly arithmetic devices. While it is true that machines were first built to carry out repetitive arithmetic operations, they are capable of other, nonnumeric tasks. The essence of the computer is the manipulation of symbols—it is only a historical accident that the first application involved numeric symbols. This incorrect notion of the computer as a strictly numeric device results in the inability of many to conceive of the computer as a device exhibiting intelligent behavior, since this would require that the process be reduced to a numerical one. The reaction of many people to statements about intelligent behavior by machines seems to indicate that they take such statements to imply complete functional equivalence between the machine and the human brain. Since this complete functional equivalence does not exist, such people believe they have thereby debunked intelligent machines. Their conclusion is incorrect because this equivalence was never implied. Intelligent behavior on the part of a machine no more implies complete functional equivalence between machine and brain than flying by an airplane implies complete functional equivalence between plane and bird.

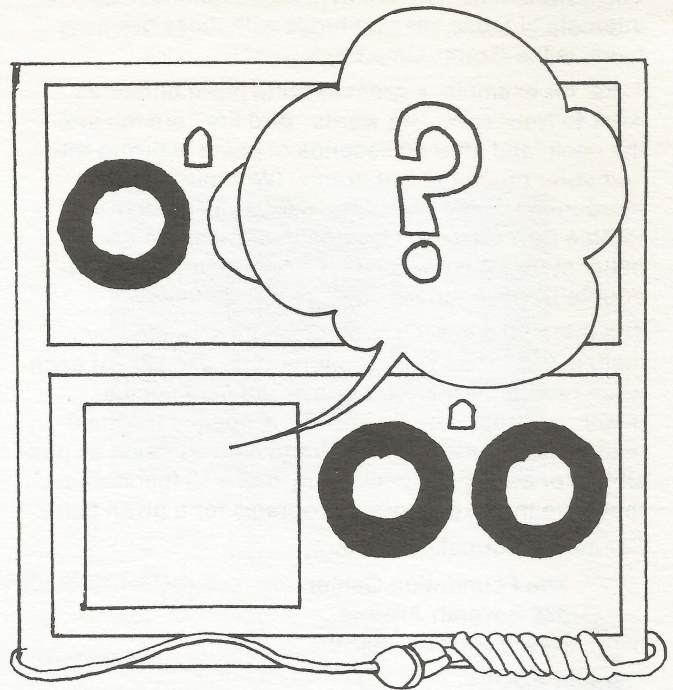
The concept of comparing the behavior of men and machines in an n-dimensional continuum recognizes differences as well as similarities. For example, a common argument against machine intelligence is that the brain is a living thing—the machine is not. In our continuum we simply recognize the dimension of living and note that machines and men occupy different positions on this dimension.

Is It Possible for Computers to Think?

No—if one defines thinking as an activity peculiarly and exclusively human. Any such behavior in machines, therefore, would have to be called thinking-like behavior.

No—if one postulates that there is something in the essence of thinking which is inscrutable, mysterious, mystical.

Yes—if one admits that the question is to be answered by experiment and observation, comparing the behavior of the computer with that behavior of human beings to which the term "thinking" is generally applied.



The two negative views are unscientifically dogmatic. The positive, or empirical, view is supported by scientists who point out that there exists a continuum of intelligent behavior and that the question of how far we can push machines out along that continuum is to be answered by research, not dogma. One might add a further qualification: to assert that thinking machines are possible is not necessarily to assert that thinking machines with human capabilities already exist (or even that they will exist in the near future).

What, then, is the goal of artificial intelligence research? It seems to be this: to construct computer programs which exhibit behavior that we call "intelligent behavior" when we observe it in human beings. Because this research area is still in the formative stage, many different research paths are being explored and a wide diversity of results have been produced.

But Doesn't a Computer Do Exactly What It Is Told To Do and No More?

Commenting on this familiar question, a well-known researcher in the field had this to say:

This statement—that computers can do only what they are programmed to do—is intuitively obvious, indubitably true, and supports none of the implications that are commonly drawn from it.

A human being can think, learn, and create because the program his biological endowment gives him, together with the changes in that program produced by interaction with his environment after birth, enables him to think, learn, and create. If a computer thinks,

learns, and creates, it will be by virtue of a program that endows it with these capacities. Clearly this will not be a program—any more than the human's is—that calls for highly stereotyped and repetitive behavior independent of the stimuli coming from the environment and the task to be completed. It will be a program that makes the system's behavior highly conditional on the task environment—on the task goals and on the clues extracted from the environment that indicate whether progress is being made toward those goals. It will be a program that analyzes, by some means, its own performance, diagnoses its failures, and makes changes which enhance its future effectiveness.

Similarly, it is wrong to conclude that a computer can exhibit behavior no more intelligent than its human programmer and that this astute gentleman can accurately predict the behavior of his program. These conclusions ignore the enormous complexity of information processing possible in problem-solving and learning machines. They presume that, because the programmer can write down (as programs) general prescriptions for adaptive behavior in such mechanisms, he can comprehend the remote consequences of these mechanisms after the execution of millions of information processing operations and the interaction of these mechanisms with a task environment. And, more importantly, they presume that he can perform the same complex information processing operations equally well with the device within his skull.

Conclusion

Thinking of the quotation of da Vinci's at the start of this article, one might paraphrase it as follows:

When men understand the natural laws which govern the flight of a bird, man will be able to build a flying machine.

While it is true that man wasted a good deal of time and effort trying to build a flying machine that flapped its wings like a bird, the important point is that it was the understanding of the law of aerodynamic lift (even though the understanding was quite imperfect at first) over an airfoil which enabled men to build flying machines. A bird isn't sustained in the air by the hand of God—natural laws govern its flight. If man gained an understanding of the processes of aerodynamics, may he not also obtain an understanding of the information processes of the human brain?

And then, once these processes are understood, there is no reason why man won't be able to duplicate in a computer the powerful facilities of association, recognition, and indeed, thinking of the human brain.

This article was adapted from the introduction to Computers and Thought, edited by E. A. Feigenbaum and Julian Feldman, and an article from the same volume entitled "Attitudes Toward Intelligent Machines" by Paul Armer.

CAUSE

CAUSE, the College and University Systems Exchange, is an organization of almost 200 American and Canadian colleges. CAUSE serves as a clearing-house for member-developed and donated programs, system designs and technical information.

The annual CAUSE conference was held December 3-5 in New Orleans and was attended by more than 300 college data processing personnel. In addition to the formal presentations, the conference provided college computer users with the opportunity to trade experiences, war stories, and even software.

DIGITAL participated actively in this year's conference and sponsored a formal session. "Minicomputers in College Administrative Applications" was presented by Alice Peters, from DIGITAL's Education Marketing staff in Maynard, and Lloyd Michaelsen, Data Processing Manager at Wheaton College, Wheaton, Illinois. Their talk covered the increasing use of minicomputers in university administrative data processing and devoted particular attention to the outstanding financial and student administrative system implemented on Wheaton College's PDP-11 RSTS System.

In addition to the annual conference, CAUSE also helps university data processing staffs by maintaining a library of donated programs which are available to members of CAUSE for only the reproduction cost. Almost 500 programs—which run on a wide range of hardware—are available, including such administrative tasks as student scheduling, inventory control, general ledger, alumni records, and general ledger accounting.

For further information about the CAUSE organization, contact Charles Thomas, Executive Director, CAUSE, 737 Twenty-Ninth Street, Boulder, Colorado 80303.



COMPUTER SCIENCE CONFERENCE

It was possibly the largest collection of people ever to meet in one location specifically to discuss computer science. Well over 1000 individuals gathered in the Detroit Hilton on February 12, 13, and 14 for the Second Annual Computer Science Conference, sponsored jointly by the Association of Computing Machinery (ACM) and the National Science Foundation (NSF).

Hubert Dreyfus from the University of California at Berkeley captivated a packed conference hall with his ideas on "The Ever Incomplete Robot," a discussion of his controversial view and research in the field of artificial intelligence (see article this issue). Anthony Ralston, President of ACM and a professor at the State University of New York kicked off the meeting with his thoughts on "Computer Science Research and Teaching." Other invited speakers included Juris Hartmanis from Cornell speaking on "Complexity Theory," Thomas Cheatham from Harvard on "Extensible Languages and Automatic Programming," and Frederick Brooks from the University of North Carolina on "Computer Graphics."

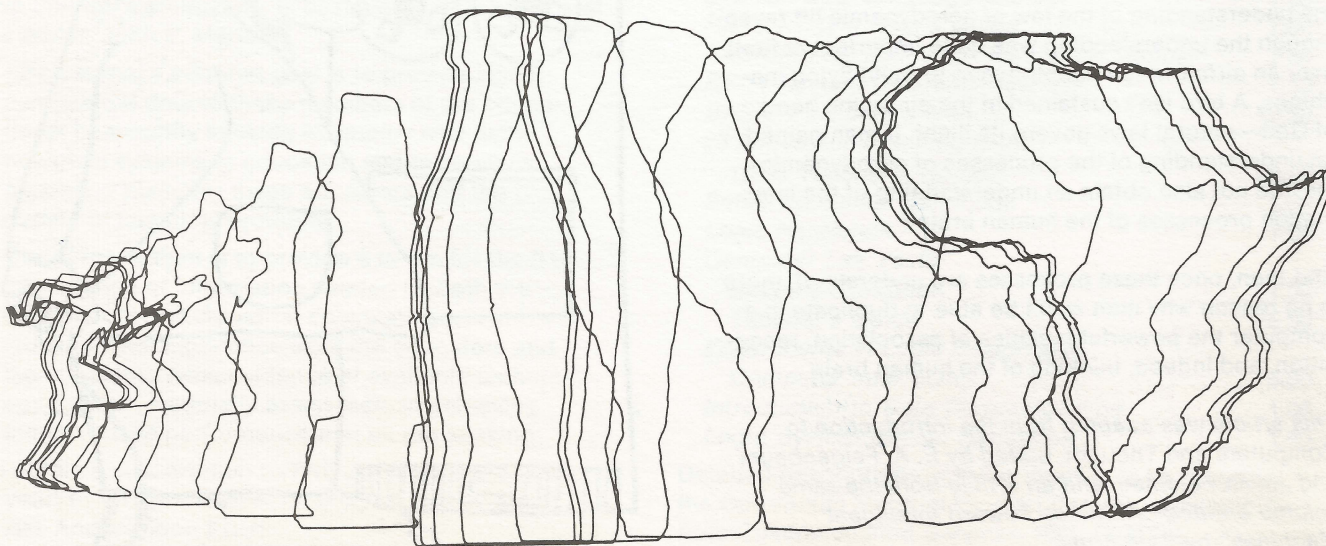
Among the most interesting educational applications described was Professor Kavanagh's description of a computer laboratory at the University of Saskatchewan designed specifically to teach minicomputer concepts. According to Kavanagh, "It is increasingly evident that a good curriculum must include exposure to the mini, both as a standalone system and as part of a larger system of computers of all sizes." A PDP-11 provides the focal point in the laboratory, which Kavanagh says provides for a "broad range of project complexity." He also stated that "the boom in minicomputer applications has caught some computing science departments short of facilities and methods for exposing the undergraduate to the mini and its uses," but offered encouraging words by noting that "a curriculum organized around larger machines can be related to the mini easily and flexibly."

"Larger" machines were by no means neglected at the Conference. George Holmes from the University of Pennsylvania and Norman Sondak from Worcester Polytechnic Institute described their schools' development of a fast FORTRAN compiler called MEDFOR on the PDP-10. MEDFOR is designed to resolve the many conflicting design criteria in the development of an ideal compiling system.

Elliot Koffman from the University of Connecticut described his use of a PDP-9 in "Generative CAI" in computer science education. "Generative CAI" systems, explains Koffman, are systems "capable of generating problems for students and deriving and monitoring the solutions to these problems." The PDP-9 teaches students how to construct a combination or sequential logic circuit using standard integrated circuits. The student's logic circuit is automatically interfaced to the PDP-9 and tested; the PDP-9 then aids the student in debugging his circuit.

Not all of the attendees at the Conference were from universities. A considerable amount of pure research in computer science is being done in the governmental and industrial sectors, a point which was illustrated by Irwin Lewis. Lewis is from Brookhaven National Laboratory and is exploring the development of a "universal controller" under the auspices of the U.S. Atomic Energy Commission. Lewis is utilizing a PDP-8 to emulate a large computer, another PDP-8 to emulate a peripheral device, and a PDP-11 to serve as a controller.

When the Conference was complete, over 400 people had described their work in computer science. If it proved nothing else, the Conference demonstrated that those "computer scientists who do futuristic things with mysterious machines" are no longer the rare species they used to be!



THE EVER – INCOMPLETE ROBOT

Hubert Dreyfus, Berkeley's Jules Verne and author of What Computers Can't Do, reported at this year's Computer Science Conference on the progress in developing an "artificial intelligence."

A machine that thinks? A robot with reason? A frightening thought to some, perhaps, but it's not entirely science fiction. According to Hubert Dreyfus, "if thinking is viewed as a continuum, and machines can be pushed out on that continuum, then machines should soon be as intelligent as we."

Dreyfus discussed his thoughts at the Computer Science Conference held in Detroit on February 13. "Although human beings are not digital computers, they are proof that material things can reason," claimed Dreyfus as he launched into his discussion.

Dreyfus described two projects which he saw as one of the most successful and one of the least successful explorations into artificial intelligence. Among the most successful, he reported, is Samuel's checkers program. The program now acts as a "Class C" checkers player, better than many of our readers, but not as good as some.

Among the least successful attempts to produce artificial intelligence, said Dreyfus, have been our efforts at producing a fully automatic, high-quality language translator. A Presidential Advisory Committee studying the possibilities for such a translator reported that such an undertaking was "not possible in the foreseeable future." The Committee recommended that projects for producing a language translator not be funded, which, according to Dreyfus, "certainly made it 'not possible in the foreseeable future!'"

Dreyfus then analyzed how the two attempts differed. "Checkers is a game," stated Dreyfus. As a game, it is "cut off" from life—what is relevant and what is irrelevant is settled by the rules of the game. But a language translator must not only store all the facts, it must also have the *significance* of the facts. "Johnny lost his coloring book, but then he found it in the pen," might be a very ambiguous statement to a computer, since the machine would have difficulty determining whether "pen" referred to a writing instrument or a place for young children to play. It is this problem of determining the meaning of words from their context which has doomed automatic language translators to date.

"The problem in artificial intelligence is to make explicit the human form of life, so the computer can share our experience with us," claimed Dreyfus. "As it is now, there is no way for the computer to understand anger, or love, or what it feels like to walk forward or backward. Because of this," he concluded, "we will still have for some time an 'ever incomplete robot.'"

ABOUT TWO YEAR COLLEGES

It is remarkable that numerous educators continue to be uninformed of the qualitative changes that community and junior colleges have contributed to education. Many still think of the two year colleges as "junior," a term which is equated with inferior. Still others envision American two year institutions as feeder schools for four year colleges, or as finishing schools, or as extensions of secondary schools.

Despite what appears to be an unforgivable lack of understanding and communication, many two year colleges have developed into fine academic institutions in their own right. These schools are an independent breed, breaking the seams of traditional education through fresh ideas, innovations and flexibility of thought. Their new vitality, indeed, makes them the major agents of change in today's education.

We at EDU believe that this "communication gap" must end. Offered here, then, are several ideas and facts about two year colleges—in hopes that the long overdue period of enlightenment may follow. DID YOU KNOW?—

- There are 900 public two-year colleges in the U.S. with 2.8 million students. Two year colleges have enrolled an unprecedented number of students. In fact, one third of all college students are matriculating in these institutions.
- In 1972-3, two year colleges experienced a 7% increase in enrollment, despite a 50% decrease in building construction.
- Two year colleges, though offerings differ from school to school, offer a comprehensive program of academic and career education, general education, adult education, counseling, and services to the supporting community.
- These institutions are an integral part of the community. Classes convene in the most unusual places (churches, warehouses, YMCA's); unused buildings and facilities are being used again. Urban renewal is being sparked, and the community is being effectively served in many locations.
- Forward thinking innovations in learning are occurring—individual education is a reality. School hours are flexible, grading is nonpunitive and transferring is untraditionally easy.
- Students are actively involved in their own learning process—they play an active decision-making role in their futures. In addition, students act as tutors, recruiters and counselors—peer involvement is working.
- 85% of the growth in higher education is occurring in two year colleges.

Is all of this an indication of the nature of things to come?

VOCATIONAL GUIDANCE FOR EVERYONE

John L. Holland
Johns Hopkins University

Although EDU does not ordinarily review journal articles, this one seemed to merit exceptional treatment for reasons of both style and substance. Dr. Holland reviews the current state of career guidance, discusses the nature of career development as the essential basis for a vocational theory, and presents some practical guidance plans drawing both from this theory and from his own experience. Also included is a fairly extensive bibliography of career guidance source material.

The excerpts below are taken from only the introductory survey of the present scene in the career-guidance field. We hope that they will serve to convey the author's provocative style and to encourage you to read the complete article. You can find it under the above title in the January 1974 Educational Researcher, pp. 9-15.

Vocational assistance comes at all age levels and in many forms (vocational counseling, vocational placement, vocational awareness, exploration and experience programs, career seminars, group counseling, etc.). In addition, such help is offered by people with a great range of talent and training—from none to magnificent.

Some unknown proportion of these services are helpful. At the same time, many—maybe the majority—share a common core of deficiencies:

1. They reach only a small proportion of the populations who need such services. For example, the ratio of counselors to students in high schools is too small to result in any marked effects. In colleges, less than 10% of the students use the counseling center.
2. The cost of service is usually high, because most services involve person-to-person treatments.
3. The constructive influence of vocational services is relatively low. When this apparent fact is coupled with the high cost of service, the cost/benefit ratio is high.
4. Agencies and educational institutions that sponsor service rarely sponsor research and evaluation. If they do, they rarely respond to evidence for the need for improvement.
5. The people who train counselors are conservative and usually addicted to personal relationships as the method of choice. Consequently, they tend to reinforce expensive and traditional practices in their students.

These generalizations are not new; they are only stated here in less felicitous form.

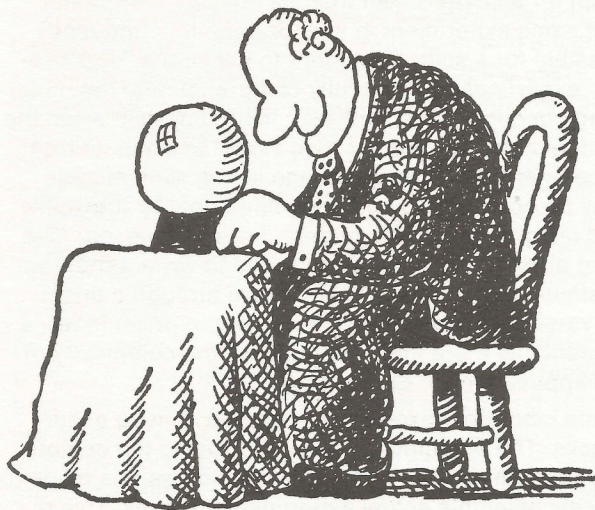
Solutions for coping with the present state of affairs also come in many forms, although a few programs or ideas are more prominent than others:

1. *Vocational Information or Guidance Systems.* These ad hoc collections of computers, terminals, microfiche readers and booklets are called systems, although their only integration is usually a kind of tasteless eclecticism. As a source of vocational assistance, these systems are, with a few exceptions, inordinately expensive, impractical (cannot serve many persons at a single time in a routine way), and require a large initial investment.
2. *Special programs for special populations.* Currently popular groups are women and persons in mid-career crises. It is grossly impractical and divisive to continue to create special programs for special groups. A better strategy would be to modify standard programs only when it is clear that standard programs do not work.
3. *Curricular materials.* These are to help students explore the world of work at a young age, although the long-range efforts of such programs are not known. Unfortunately, this burden falls on teachers who already are responsible for mental health, discipline, and community chest, in addition to the three R's.
4. *Career Education.* This new brand of education is to careers what milieu therapy has been to mental hospitals. It is an attempt to orient and manipulate students for their own good from kindergarten through the 12th grade. Its comprehensiveness and complexity are not yet in full bloom. If it flourishes (and Federal money should help it do so), in the next few years an Avocational Education Association may be formed to put sloth, slovenliness, and recreation back in the schools.
5. *Assorted assessment devices, tests, inventories, and informational materials.* Publishers are churning out auxiliary materials that could be labelled career, vocational, guidance or occupational. This collection of materials is characterized by its atheoretical approach, use of contemporary formats and colors, and absence of evaluative studies.

In short the practitioner has a great array of old and new tools, ideas, devices, and programs. At the same time, he appears unable to organize or integrate those resources he now has with those he might purchase,

borrow, or imitate. In desperation, he buys more of the tried and true, establishes ad hoc programs as he gets new pressures, tries to buy whole universal programs, or succumbs to the canned programs pressed upon him by state or federal agents.

How did we arrive at this current state of confusion and why do we remain in it? Vocational guidance practice and research are weakened by numerous misunderstandings, meaningless questions, and misguided forms of humanism. I will try to clarify or dispense with these barriers to better knowledge and higher quality service with a discussion organized in order of my irritation.



"Counseling Must Be Personal." The counseling and teaching professions attract a large proportion of friendly people who must love and be loved to get through the day. Consequently, they believe that other people also must have the same needs with the same intensity. As a natural corollary, many also believe that any form of vocational intervention must provide for a person-to-person situation. These beliefs have prevented any major revision of the delivery system for vocational services. Some experience and recent experiments strongly imply that most people want help, not love. In no case has an impersonal information or guidance system received a lower average rating than local counselors. To the contrary, most tests reveal that impersonal schemes are more highly rated as well as infinitely cheaper, have better attendance records, and are generally more dependable. In short, there is ample empirical evidence to support more impersonal approaches for the solution of vocational problems.

"Vocational development must be in the whole curriculum from kindergarten through the 12th grade."

The major assumptions here are tenuous: (1) It is possible to specify vocational treatments or experiences at every grade level that will produce more adaptive vocational behavior in adult life, (2) everyone needs the complete treatment from K-12, (3) the main vocational development problems are lack of occupational information and lack of information about self, (4) people cannot make up deficits, incurred as children, at older age levels, (5) vocational stages in development are well-defined, thoroughly researched, and reliable indices of vocational development, experiential or training needs. None of these assumptions has strong empirical support.

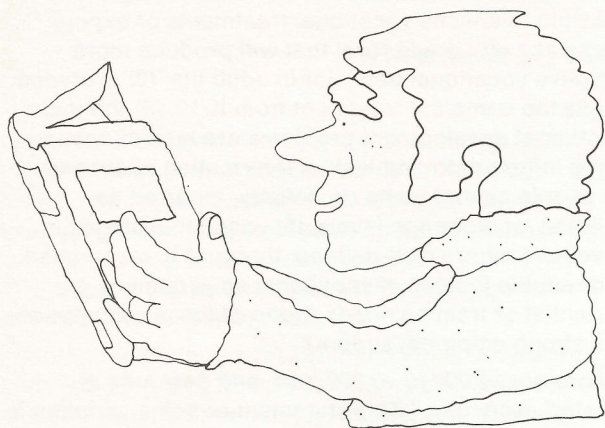
"There are 30,000 to 40,000 jobs, and new jobs are created every day." This statement or some variation is used to demonstrate the need for vocational guidance. In contrast, data analyses repeatedly indicate that there are only five to eight major kinds of work. True, there are 40,000 occupational titles, but most of these are minor variations on some main themes. Likewise, the new jobs are new all right, but they nearly always closely resemble some old jobs.

"Women need a special theory of career development." Because women belong to the human race, they probably share with men the same principles of development. Their different life styles are due largely to the outcomes of special cultural experience and are not due to the operation of special behavioral laws or principles. Men who are subjected to the same experience as women act in much the same way. For example, men and boys whose initiative and talents are deprecated also have difficulty in obtaining jobs and making a career.

"Stamp Out Indecision, Irrationality, and Anxiety." In the 1940's it came to be believed that "undecided" students are either troubled or in trouble. This belief went unchallenged for 20 years as counselors searched for undecided students and attempted to convert them. Several large scale studies, however, suggest that undecided students are like most students: they make subsequent decisions and their decisions are as sensible as anyone's. Only for a small minority does indecision seem to represent severe conflict and instability.

Irrationality and anxiety usually go hand in hand with any major decision that involves great ambiguity, risk, and long term consequences: getting married, buying a house, choosing a job, etc. Although it is desirable to become as rational as possible about such decisions, learning to rely upon one's inner impulses should not be underestimated as a useful guide in ambiguous situations. Likewise, fear and anxiety often serve to motivate people to seek counseling, information, and advice when no other personal or situational force will bring an adaptive response.

ARTICLES AND PAPERS



STUDENT-WRITTEN SIMULATIONS IN LABORATORY SCIENCE

Bruce Presley, Theodore Graham, and Edward Stehle
The Lawrenceville School
Lawrenceville, New Jersey

The laboratory in many traditional secondary school science courses is of questionable value. The theory is that it gives the student a deeper understanding of the laws of nature and an experience in the process of scientific discovery. Too often, however, little is learned, and the experiments often demonstrate concepts that are better covered in class or from a textbook. In an attempt to improve the students' laboratory experience, the science department of The Lawrenceville School, three years ago, introduced computer simulation as a supplement to some physics and chemistry laboratories.

We began by using a single terminal timesharing facility, expanded to two terminals, and finally, in the fall of 1973, installed our own four-terminal PDP-11/40 RSTS/E computer system. The advantages of the in-house system became immediately apparent: students have considerably greater access to the computer, the new system is more reliable than the remote terminals, and most important, we now have a large program and file storage capability.

Computer simulation is a technique for developing a mathematical model of a physical situation. A successful model will produce results similar to those achieved through experimentation.

We have found that student-written simulations are preferable to "canned" programs that allow the student to vary parameters without creating the initial model. Our approach requires that the students know a computer language; initially, successful simulations have been created with only a limited knowledge of BASIC. We have discovered that only a few classroom lectures on selected programming fundamentals are needed to permit the student to begin simulating experiments. As the demands of the simulations become more complex, students have usually been motivated to learn more sophisticated computer techniques on their own. While this approach is most successful with advanced students, it need not be limited to them. We have found it successful even with average ninth graders!

Our experience with simulations convinces us that a student often learns much more by writing his own program for an experiment than he does by performing that same experiment in a laboratory. It is frequently possible for a science student to perform a "successful" experiment without fully comprehending the underlying concepts. This, however, is not possible when the student must develop his own model to simulate that experiment. Another advantage is that such models may be open-ended and are limited only by the ability and imagination of the student. For example, our first term physics students are required to write a program to simulate the path of a light beam through a prism. By varying the angles and length of the prism faces, a student can more fully comprehend the complexity of the applications of Snell's Law.

Some laboratory experiments involve chance occurrences. The outcomes produced by using the computer's random number generator function can be made to closely resemble actual experimental results. This reinforces the student's awareness of the statistical foundations upon which many scientific laws depend.

Additional advantages accrued from applying computer simulations in science courses are:

- 1) safety (especially true in chemistry and nuclear physics)
- 2) reduction of the time required to perform experiments
- 3) laboratory equipment expenditures may be held to a minimum; and
- 4) the student's mathematical skills are re-emphasized and applied to other fields.

We have not, however, wholly replaced our traditional laboratory program with computer simulations. There is still value in learning how to make careful measurements and in experiencing the frustration of performing experiments which will not work properly. However, we have added a learning device that enhances the student's realization of the nature of physical laws and have introduced him to a technique that he will encounter in university and industrial research.

THE COMPUTER AND HUMAN THOUGHT

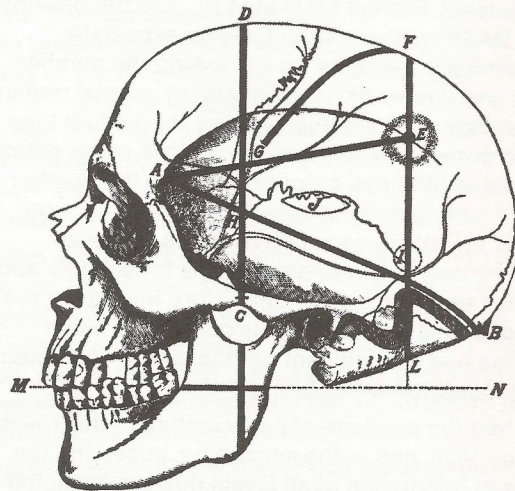
Warren Eisenberg
Director of Project LaSalle
LaSalle Academy
44 E. 22nd Street
New York, N.Y. 10003

With the next decade or so it would seem rational to state that computer science will become a part of the curriculum of almost every high school and college in the United States and, indeed, in the world. Human thinking, whether it be creative or applied, will find the computer as necessary a tool in the future as the ax was to the cave man of the past. This statement is made from personal observation as a college student, as an engineer in the aerospace industry, as a systems analyst in commercial computer applications, as a teacher, professor, and currently as director of the largest computer network of private high schools in the world—the La Salle Computer Project in New York City.

This varied background, most of which occurred by chance, had one common thread—wherever I went, whatever I was doing, a computer was either directly or indirectly involved. In college a computer printed tuition bills, transcripts, and scheduled classes. In the aerospace industry, it seemed the computer was used more than human language: trajectories of spacecraft to distant planets, earth satellite orbits, pay checks, vast calculations day and night, even personalized letters written by a computer. In the military there were computer "war-games," airborne computers, computers on ships, computers for huge management information systems, more pay checks, etc. In commerce we developed computerized ticket reservation systems for airlines, automated accounting systems for banks, and on and on, ad infinitum.

One begins to feel like Jack with the Beanstalk. Jack is at the bottom looking up and wondering where it all leads to. We are at the beginning of something relatively new and utterly fantastic. What the mating of human thought and the computer will continue to lead to is as varied as the people thinking about it.

Four years ago I became a faculty member at LaSalle Academy, a small private Catholic high school in the Lower East Side of New York. The school has about 750 students. Three years ago we introduced a small computer programming course, using a DIGITAL mini-computer, for a select group of 30 students. It was wild. The kids loved it—I loved it—the faculty was fascinated. More and more students wanted to take



the course. We expanded our equipment to accommodate 200 of our own students. Other schools heard about us and wanted to do the same. Over the last three years, La Salle Academy gradually fostered a consortium of Catholic schools in the New York metropolitan area, now numbering 25. Over 25,000 students can access our two large DIGITAL computers. Computer programming has become the most popular course at each school. The students are taught how to solve math problems, business problems, cartooning, physics problems, how the computer is used to edit newspapers, assembly language programming, BASIC, FORTRAN, etc. Our computer center operates from before sunup to sundown with students begging to come in after school for extra time.

Reports from previous graduates have been quite positive. The one great fact is how the knowledge of computer science has helped them as an extension of their own thinking. One student went into police work and commented, "Now I see how they trace license plates." Another student commented that he now knows how interest is computed on bank passbooks. Another student figured out how Gimbels and Macy's check for valid credit cards.

We extend a cordial invitation to all interested readers to visit our facilities. Indeed, it will be obvious to any visitor that the computer has become an integral part of our school and the thought processes produced therein.

COMPUTERS AND MATHEMATICS INSTRUCTION— THE THIRD GENERATION

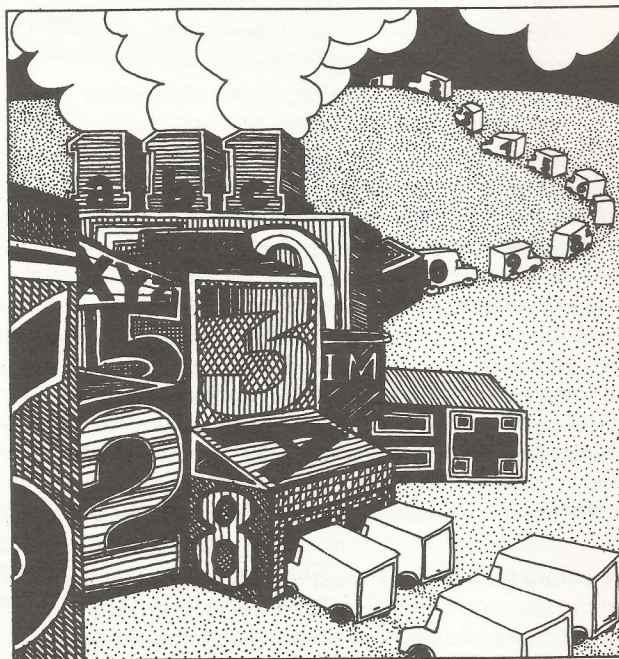
G. Albert Higgins, Jr.
Northfield Mount Hermon School
Northfield, Massachusetts

For some eight to ten years computers have been influencing education at the secondary and undergraduate level. Project MAC at M.I.T. and the development of BASIC at Dartmouth College especially accelerated the trend, for by increasing the number of users and decreasing the amount of special training required, large timesharing systems for the first time provided computing services for novices in the educational field at little risk or investment. As the number of users multiplied, there was a fertile field for mini-computer manufacturers. DIGITAL has achieved notable success in stand-alone minis for schools and, now, small timesharing systems. More important, the growth of educational materials during this same period has had a significant parallel effect on content and curricular refinement.

Having had the privilege of participating in this revolution since 1966, and at the same time, observing the changes in instruction at all levels through participation in the major conferences, I believe it is possible to trace the impact of computing on instruction through a series of cycles or generations, for instruction evolves at varying rates, with a rhythm and direction of its own.

The First Generation

Our first attempts were surely crude and inefficient in retrospect. In many cases schools used computers just to experiment. The goals were limited: learning to program and to impress students with the speed and accuracy of a computer. These goals should not be scorned, yet their achievement frequently was accomplished with sacrifice to existing mathematics instruction, for often units of computer work were artificially grafted to courses where little relation existed. Many teachers (as adults often are) were more in awe of the machines than were students, and were negative about possibilities for the computer as a legitimate teaching tool. All of us recognized the significant increase in motivation of large numbers of students and rejoiced at the by-product of mathematical inquiry. From the point of view of appropriateness of integration of computational and mathematical topics, many of us erred on the side of the trivial or the ultra-sophisticated, for practically no curriculum materials were available other than mimeographed notes from the few "experts," or worse—traditional numerical analysis textbooks designed for mathematics majors and mechanical calculators.



The Second Generation

After a few years curriculum materials became available so that each new program could benefit not only from the experience of the experimental programs of the sixties but could operate at an appropriate academic level as well. Without citing specific texts, these materials provided supplementary material for existing courses in mathematics, or for introductory courses in computer science.

The past generation of students found applications far more worthwhile than their teachers did, and this fact alone did more to convince hesitant faculty that the computer was here to stay. As a result thousands of experienced teachers attended in-service training programs and post-graduate courses, and now it is the rule rather than the exception that teacher training graduates know how to program. Many teachers found that demonstrating mathematical concepts such as limits of series, or solving for irrational zeroes, or developing calculus definitions using a computer, was not only practical, but from the student's viewpoint made the assimilation far less painful. Clearly the computer had become an adolescent.

The Third Generation—A Coming of Age

We are now just reaching a state of maturity in instruction in terms of approach and effectiveness with computers. Schools with computers now have the sophistication to utilize them in instruction where it is appropriate and more effective than traditional methods, as well as the confidence not to be obliged to experiment simply to justify budget expenditures.

Most important, as students and teachers make use of computing and are able to program, teachers of advanced courses are in a position to capitalize on the knowledge of their students without the burden of extended sessions.

To give a specific example of change of approach and effectiveness I shall draw upon one from my own textbook on functions,¹ although surely numerous others might be used. All students are reasonably proficient in programming by the time it is desirable to study the trigonometric functions, and an algorithmic method is developed to find the sine of a real number. Knuth,² in defining an algorithm as a process of transforming relevant data to desired output after a finite number of steps, recognizes the importance of analyzing the procedures involved in assimilating the mathematics itself. In a course in functions, students gain an appreciation for functions where the rule is not an equation but the result of a finite number of calculations. Short of calculus methods, there is no more satisfactory method for learning about many functions than such numerical processes.

Recently published curriculum materials in the era of "new mathematics" have introduced circular functions early, where, for example, the sine function is defined as the y-coordinate of a point on a unit circle determined by an arc whose length, x , is measured in relationship to a suitable coordinate system. Functions of angles are introduced later. Frequent comment is made about the traditional courses where right triangle functions precede functions of a real number. Also "new mathematics" de-emphasizes the study of identities. Yet the definition of sine in either approach is unsatisfactory from the student's point of view, for he has had limited ability for computing sine arbitrarily, whether the domain was angle or real number. Elementary algebra and geometry produce a few special values: at quadrants, $X = \pi/6, \pi/4, \pi/3, 2\pi/3, 3\pi/4$, etc.

For other values the teacher had to say, "refer to the tables," or worse, introduce Taylor series, which are not properly understood until calculus. Students and teachers are not, nor should they be, satisfied to accept solely a table with the promise that the values are accurate.

How can numerical methods be brought to bear? First, one should recognize that many of our familiar identities were computing devices developed by Klaudios Ptolemy (c150 A.D.) precisely to find values of the "chord of an arc," the antecedent of the modern sine. His was the first recorded table of "sines." The identities resulting are:

$$1. \sin(A \pm B) = \sin A \cos B \pm \cos A \sin B$$

$$2. \sin(x/2) = \sqrt{\frac{1 - \cos x}{2}}$$

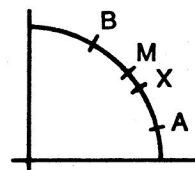
$$3. \cos(x/2) = \sqrt{\frac{1 + \cos x}{2}}$$

Today these are proven analytically on the basis of the coordinate definitions, but once proven, form the basis

of a crude algorithm which is far more acceptable to students than Taylor series. The algorithm to find the sine of X is as follows:

1. Find the sine (and then cosine) of any two points on the unit circle which bound the point determined by the arc of length X .
2. Compute the sine (and then cosine) of M , the midpoint of arc AB , using the identities shown and the fact that

$$\sin M = \sin \left[\frac{A+B}{2} \right] = \sin \left[\frac{A}{2} + \frac{B}{2} \right]$$



3. Using the usual technique of interval reduction in solving for irrational zeroes using bisection, choose M as the appropriate replacement for A or B .
4. Continue the process until the arc length is sufficiently small ($|A - B| < \epsilon$), or the values of $\sin M$ remain constant for a sufficient number of decimal places.

Such an algorithm is a particularly appealing adaptation of computing techniques introduced in algebra. Identities are used for the purpose for which they were invented, and the student may calculate them at will. (See Reference 1, pp. 280-287 for more extensive description of this algorithm.)

Finally, iterative methods are representative of one of the usual modes of scientific inquiry to make approximations of concepts which are refined in stages to one empirically acceptable. As such they may be equally important as the analytical methods of classical mathematics, so that our courses can become truly relevant through computation.

¹Higgins, G. A., Jr., *The Elementary Functions: An Algorithmic Approach*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1974.

²Knuth, D. E., *The Art of Computer Programming: Fundamental Algorithms*, Addison-Wesley, Reading, Mass., 1969.

It is unworthy of excellent men to lose hours like slaves in the labor of calculations.

—Gottfried Wilhelm Leibnitz, 1871

ON THE SCIENCE OF COMPUTING

Rev. Dom Geoffrey Chase
Portsmouth Abbey
Portsmouth, Rhode Island

It may help to lay one's cards on the table. My early education was classical, mostly Latin and Greek and math. I took my B.A. in math, my later degrees in theology, with some specialization in the languages of the ancient Near East and in the writings of the Greek Cappadocian Fathers, who, of course, were Persians, since they lived in Turkey. St. Gregory Nazianzen cites the Russell paradox in one of his sermons, which proves that prophecy was still around in the 4th century. The intellectual center of Greece at this time was in Egypt.

So my natural frame of reference is in some ways more that of Plato, Aristotle, Augustine and Aquinas than that of Dewey, Durkheim, or Skinner. If the former are little read today, least of all by clergy, they may at least stand as reminders that there are other options and other frames of reference than those on which the Yale Class of 1952 was raised.

The computer is now irreversibly entering our intellectual world on a large scale. A great deal of the conventional wisdom is already obsolete, and more will follow. Perhaps the worst thing we can do is to try to tame the beast, to make it the docile servant of an existing academic order. Alas, it doesn't know its place! And the attempt so to limit it is in essence the attempt to cheat our own students. The second worst thing is to worship the computer with the kind of incomprehension (and implied disinterest) usually reserved for deities. One of those ancient writers observes somewhere that there are hazards in bowing down before the work of one's own hands.

So we are left with the necessity of understanding the computer. It is an artificial intelligence, a small model of the natural intelligence that designed it. There is much to be said for understanding its hardware instruction set and the underlying electronics—of this more later—but what we are really setting out to do is to understand intelligence itself and how it operates. In studying the computer, we are by indirection studying ourselves—an occupation commended to us by the Education Department at the Delphic Oracle.

In reading Knuth's *Art of Programming*, the Baedeker of this new world, one is struck by the appearance of names from an ancient past: Abu Ja'far Mohammed ibn Musa al-Khwarizmi, to whose work "al-Jabr..." we owe "algebra" (though it wasn't about algebra very much) and to whose locative last name we owe "algorithm," which our pedants Hellenized to "algorithm," by much the same false analogy that put the useless "s" into "i(s)land." Then there are all those Chinese, such as the author of the "Great Generalization," known to us the Chinese Remainder Theorem.



Euclid turns up; his algorithm for the g.c.d. is still respectable. Leonard of Pisa (Fibonacci) wrote while Chartres was a-building; his remarkable learning is not unrelated to that concretization in stone and glass of a geometer's vision of Heaven. Pythagoras is sculpted over one of the portals, and would have approved. It would seem that the "analytical engine" points both forwards and backwards in time. The classical love of form and structure, that strange insistence of Euclid on proving the constructability of a figure before discussing its properties, seem suddenly rather modern.

The computer makes its entrance upon our scene at what is probably the end of an era. We have heard the cry "Wolf, Wolf!" often enough to become skeptical of apocalypse. A hundred years ago, the problem of a particular kind of vehicular pollution politely referred to as "mud" was exercising the minds of contemporary prophets. One projection had London and New York buried several miles deep by the late 20th century!

It didn't happen. But apparently there are some real wolves at our door today, all the same. The next generation is unlikely to have the cheap energy, the rapid and private transport, the indefinite mobility, physical and social, upon which our own way of life has been based. One hopes that at least they will retain their personal freedom, but there is no surety even of that. *Time* magazine not so many years ago had an issue

devoted to proving the virtues of waste and the viciousness of thrift—it slowed down the economy. These values are now reversed, and we are in consequence badly out of communication with our recent past, and by the same token more open to our earlier heritage. The man-made world, the universal Astrodome, which has been for long both our ideal and our idol, seems less feasible, and far less desirable, than it used to. One can play games with Nature, but cannot escape her laws; ask anyone who lives by Lake Erie.

What we must also face up to, and what to date we have not faced, is the problem of human nature. We are accustomed to considering ourselves as self-made beings; could there be here, too, limits, even laws? In our man-made world of the last two centuries, the use of the term "reason" has been restricted to ever-narrower subsets of human discourse. Not only is "God is transcendent" incapable of meaning, truth or falsity, but so also (say some) are value judgments of whatever sort—e.g., "Genocide is immoral." If this is really so, then the mind has nothing to offer us in our search for meaning and humanity. The only authentic life-style is contained in the one and greatest commandment: "Blow your mind, baby."

What has this to do with computer science? Everything, I'm afraid. If we undertake to study intelligence itself, we must be prepared to justify our study. I do not think the young are going to take it on faith much longer. Some that I know have already decided that thought is evil, and the man of reason the enemy of man.

I reject, for my part, this antithesis between the "hard" knowledge of the positivist (generally propped up by some metaphysical assumptions that are as "soft" as they come) and the rest of human perception. The medievals distinguished reason from intellect as a valuable, but partial, subset of a larger whole. Any deductive science, for example, assumes as defined certain undefinables (point, line) and assumes as proved certain unprovables perhaps the parallel postulate); otherwise there is nothing to deduce from. But which assumptions provide the best model for the task at hand? Here intuition, imagination, and experience come into play. It's curious that both Aquinas and Einstein saw that any model of the solar system can be true only in a relative sense; by contrast, both Galileo and his enemies thought that theirs was the real universe! A. and E. were far more aware of the nature, and the limits, of intelligence itself than were the telescope-maker and his enemies in the civil and ecclesiastical bureaucracy.

It is this deeper sense of the nature of intelligence, human or artificial, that we must teach. The computer, the analytical engine, could be used to sharpen the awareness of structure, of process, of logic operating within a finite framework—and of their limitations. But I fear very much that it will be used instead to avoid hard questions by escaping into the analysis of vast amounts of dubious data. The machine that gives us such enormous power gives little by way of guidelines as to its use. These we must find for ourselves, or be destroyed by our own handiwork.

The second grade of a neighboring parochial school came in one day to meet the "monster." Besides a few games—my version of "Matches" brags insufferably when it wins—there was a program which ran somewhat as follows (user input is italicized):

PLEASE TYPE IN YOUR NAME? *JOHN*
THANK YOU, NHOJ—THAT'S RIGHT, ISN'T IT
(Y OR N) ? *N*
SORRY, JOHN, GUESS I GOT IT BACKWARDS. BUT
YOUR NAME IS OUT OF ORDER!
ISN'T THIS BETTER: *HJNO* (Y OR N) ? *N*
NO?? WHAT KIND OF A MACHINE ARE YOU,
ANYHOW ???!!

Well, two things happened that I had not anticipated. One girl was named Amy! And one boy, after taking his turn, stood in line—second-graders line up for everything from drinking fountain to toilet—until he was again at the keyboard. This time he entered his name *backwards*.



Now there is a discovery—and a programmer. Perhaps he would not have grasped that a double inverse transformation is simply the identity if he hadn't met the problem in a computer context. If so, then computers are needed even more in grammar school than in high school.

The writings of Professor Seymour Papert and the "turtle" people are openly hostile to the "new math," especially on the elementary level. In part, I agree. I don't like too much logical quibbling at a level where experience is still so limited. Certainly something has slipped in the logical, verbal, arithmetic, and cultural formation of the students I have taught these last few years, and they are scarcely the most disadvantaged. But I do not agree with the bias expressed against integers; on the contrary, since the only numeric or logical states expressible in a digital machine are integral, I think them more important than ever. I wonder if during their early years children shouldn't meet:

- (a) Squares and square roots. $SQR(5)$ might be described simply as "between 2 and 3", which introduces the floor and ceiling functions (in BASIC, $INT(X)$ and $-INT(-X)$).
- (b) Primes and prime factors. Simple short division can get you started on this.
- (c) Modular arithmetic. 11 o'clock times 2 is ... ?
- (d) Simple sequences and series—the sum of the 1st five odd numbers.
- (e) By way of fun and games, Russian Multiplication (which is really Babylonian, and dates from about the same year that the earlier part of Stonehenge was begun, namely 1800 B.C.), or subtraction by complementation on 9 or 10, as an illustration of the effects of a finite format upon arithmetic.
- (f) I bet that they can discover factorials by themselves—I know I did. Arrange 3, then 4, tin soldiers in all possible orders.

I have a hunch that the algorithms we teach kids in arithmetic need a good and thorough review. How many of us can actually do long division?

I find high-school students seem to fall into three categories where the computer is concerned. The same individual may, and often does, migrate from one to another as he grows and his interests change:

- (a) These are the people who want results and nothing more. They will do their chemistry or physics homework, painfully, calculation by calculation, rather than write a program; or, they'll learn how to load in a canned game routine and then play it by the hour. What they most want is to minimize their input. The programming language excites little interest, the processor none. By "computer" they usually mean Teletype.
- (b) The "clever" ones, gadgeteers essentially. They learn to manipulate software or console switches so as to get what they want. Typically, they write games, carefully borrowing as much as possible from books. Moderately competent technicians, the structure of a problem interests them only if there is a "bug" they can't see how to solve. ("Bugs," incidentally, are probably the most educational thing a student ever writes.)
- (c) This is the student, often not a scientist, who somehow grasps the nature of the processor itself. Machine language work is almost the only way this seems to happen. At this point the student needs some help, a few references—and then there is a knowledge explosion and you start learning from him.

Of the (b) students and of the (a) students as will spare the time to listen, I ask two questions: "what do you want to do?", and "how would you set about doing it if there were no computer here?" I have seen hours spent on line by someone who hadn't the foggiest notion what he wanted the processor to do for him; the whirr of an ASR-33 was substituting for thought, the growing yards of paper for perception. There is only

one cure for this: grab by one ear, drag into empty, silent room, and ban the computer entirely until the lad can tell you what he wants it to do.

The better (b) students will probably want to learn FOCAL as well as BASIC. Excellent; this will clear their heads of the idea that all flip-flops graduated from Dartmouth. Lure them into the 8K FORTRAN and let them see the listing of the object program. The distinctions between assembler, compiler, and interpreter now begin to fall into place quite naturally.

The (c) student—the good one—will gravitate to machine language if anyone around is competent to start him off, and even if not, sometimes. Someone on the faculty really must acquire this skill, for the sake of his own understanding even more than for any possible use in teaching. Until this point is reached, you're essentially a captive customer.

For those many classes that will use the machine simply to calculate answers to math problems, try to insert at least the following:

- (a) Add $1/7$ to itself for a total of 700 times. The answer is not 100.
- (b) Do the same with $1/8$ and 800. The answer is 100, exactly. Why?
[One can explain this without going into binary floating-point, though of course it would be nice. Pretend the machine is a 7-digit decimal device internally.]
- (c) Use the machine to "prove" the convergence of $1 + 1/2 + 1/4 + 1/8 + \dots$
- (d) ... and of the harmonic series, or a disguised subset, as well!
[What can be said about such "proofs"? Would having a private PDP-10 or ***370 really change this situation?]

About faculty:

- (a) Expect little comprehension and a good deal of hostility, overt or covert. There are few things we teachers fear more than showing our ignorance, even to each other.
- (b) Lure them with goodies that will average and rank their classes, or (as here) with a universal German vocabulary/grammar program that allows for answers of appreciable length.
- (c) Try to give those faculty (who will, or should, use the machine) time when they can make their inevitable mistakes and not feel humiliated in front of their own students. We know, by and large, a lot more than our students, but they are by far the quicker learners!

Remember that a complete shift of thinking is involved here. How, rather than whether, comes to the fore. This contradicts much of my own education, especially the math part, and probably much of yours, *lector benevole*. It will take time and patience to adjust to yet another revolution, but it has to be done.

TEACHING ASSEMBLY PROGRAMMING WITH SPACEWAR

D. E. Wrege and D. S. Harmer
School of Nuclear Engineering
Georgia Institute of Technology
Atlanta, Georgia

With the development of low cost, flexible minicomputer technology it has become increasingly important that students in technological disciplines understand how to use small computers for their major subject applications. The depth of understanding required is generally not that of a hardware designer or software systems engineer, but rather a level of knowledge needs to be attained so that these students can communicate with computer specialists.

In order to satisfy this need a computer engineering minor is being developed at Georgia Tech; this course-work uses a "hands-on" approach, introducing the students to both hardware and software concepts. The program is aimed at the graduate level engineering student with little or no prior training in digital design or computer science.

The first course in the series is aimed at an in-depth understanding of hardware and interfacing and hardware-software "trade-offs" (*Experiments in Logic Design and Computer Interfacing*, D. S. Harmer et al, DIGITAL, 1971).

The second course in this series deals with assembly level programming techniques oriented primarily toward "on-line" or "real-time" applications. With the current trend in small computer software, the use of machine language programming as an approach to solving problems has been de-emphasized. High level languages with real-time capabilities have been developed for many minicomputers, and the trend is to use these languages for real-time applications. What

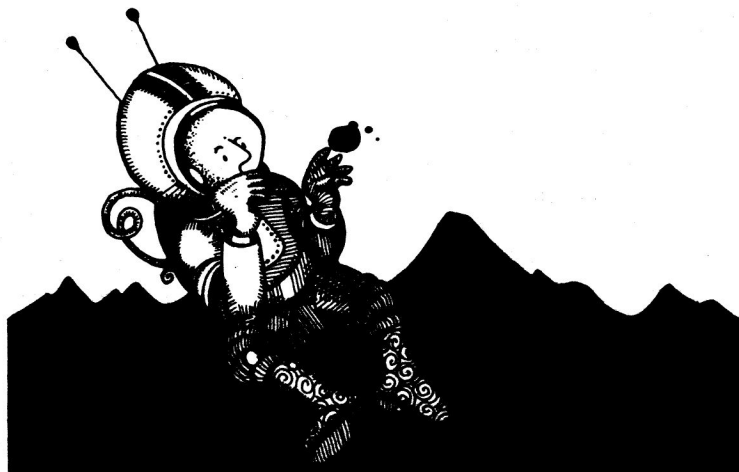
is often overlooked is the fact that some assembly (machine language) programming needs to be done to tailor the high level language to the real-time application undertaken. Once this is accomplished, the majority of the programming task may be accomplished in the high level language itself. Unfortunately, although the machine language portion of the coding effort may be only 5-10% of the total programming effort, a comprehensive understanding of this subject is required to accomplish the desired end.

It was with the above motivation that the introductory course in software systems was oriented toward machine language coding techniques. A DIGITAL PDP-8 is used and is especially well suited for this instruction, due to the small instruction set and simple addressing schemes. Similarly the I/O structure is uncomplicated and easy to understand.

The first attempt at teaching this material proved less productive than anticipated. After a great deal of retrospective analysis the teaching approach was modified, and a vast improvement in student response was experienced. The purpose of this article is to relate our findings and discuss this new approach.

The value of a hands-on approach to learning was realized. In addition, the fact that programming techniques was best learned by writing programs was also understood. What was *not* realized was the importance of program content. For example, a student is better equipped to undertake real-time programming tasks with 50 hours of non-real-time program assignments

Continued on p. 37



APPLICATION STORIES



HERKIMER COUNTY

Herkimer County Community College is a two-year institution operating under the supervision of the State University of New York and sponsored by Herkimer County. Founded in 1967, HCCC grants the Associate in Applied Science, Associate in Arts, and Associate in Science degrees.

The college practices a full-opportunity admissions policy. Program placement and admissions counseling are based on the multiple factors of high school achievement, objective test data, interest, maturity and motivation. Such a policy places critical demands upon the instructional process; it must provide a range and depth of methods not usually required where more rigorous admissions selection is practiced.

In the belief that a computer system could help to expand our instructional range, the college applied for a Title VI-A Higher Education Act grant. The following expected benefits were stressed in our proposal:

1. Individualization of instruction—Operating under an "open-door" admissions policy, the college finds great diversity in the achievement levels of students. The acquisition of this computer would provide the hardware necessary to develop instructional programs that will meet the needs of these heterogeneous classes.
2. Flexibility of learning modalities—The addition of this equipment would provide yet another method by which learning may be facilitated. It allows for a greater spectrum of techniques in reaching the naïve learner.

3. Increased sophistication in problem solving—The rapid expansion of knowledge and techniques, especially in the math-science area, requires the ability to locate and identify new problem solving procedures. The use of data processing equipment provides increased flexibility and sophistication in this area. To successfully conduct instruction in such areas as population studies, gene frequency and metabolic studies, there exists the need for sophisticated instructional methods like computer simulation.

In June, 1972, HCCC was notified by the Department of Health, Education and Welfare that it was the recipient of \$12,945 from the Office of Education, Bureau of Libraries and Educational Technology. Our Edu-System 25, consisting of a PDP-8/E with 12K of core, TC08/TU56 DECTape, and 5 local teletypes, was installed in November of 1972.

The physical arrangement of this system is quite special (possibly unique in New York State community colleges). The Business and Math-Science Divisions share the same building. The computer and three teletypes are located in a data processing laboratory on the first floor. The remaining two teletypes can be used in any of the four science laboratories (2 biology, 1 chemistry, 1 physics/geology/astronomy) on the second floor. Each lab has an outlet which permits the teletype to be hard-wired to the CPU. Other distributions of four of the five teletypes are possible.

More than 25% of the college's students were involved in using the computer in the initial year. A series of faculty seminars helped to raise pedagogical interest in the computer, which is reserved exclusively for instructional applications. Due to the recent installation of the system, no formal evaluation of the program has yet been completed. However, in addition to the instructional improvements discussed in our grant request, we feel that the college has also realized the following additional benefits:

1. An improvement of the computer science course, which previously had afternoon access to a time-shared computer at Griffiss Air Force Base via one teletype.
2. A more favorable position for HCCC graduates who are transferring to four-year institutions, where the use of a computer system is commonplace.
3. Exposure of students in the Data Processing curriculum to a time-shared computer. These students previously used only a batch machine.
4. A more competitive position relative to area institutions in the recruitment of students.

Submitted by: David P. Liebchen
Herkimer County Community College
Herkimer, New York 13350

A COMPUTER SCIENCE SUPPORT LABORATORY

G. W. Hinrichs, III and W. J. Lennon
Computer Science Department
Northwestern University
Evanston, Illinois 60201

One of the resources of the Computer Sciences Research Laboratory (CSRL) at Northwestern is the "teaching lab," a PDP-8 based timesharing system that supports eight machine language programming consoles. This system is used to support the programming laboratory portions of the EE 801 and CS 802 courses.

The system was designed and implemented with a projected user population of about 40 students per year. During the past year, 140 students were enrolled in EE 801 and 64 were enrolled in CS 802. This load at time pushed the system to its limits even with a sign-up system for allocating system time among the users. Interest in the two courses is growing within the Technological Institute and the College of Arts and Sciences. We now plan to offer EE 801 and CS 802 all three terms and limit the enrollment in each course to 40 students per term. The new policy will result in a load of up to 80 students a term, which in turn will almost certainly result in a great deal of time pressure on the system.

As knowledge of and interest in the teaching lab has spread, many suggestions for expansion of its features have been offered. Among the most interesting of these are incorporating a high-level language capability and providing the student with the capability of writing programs that directly control input/output operations. Either, or both, of these capabilities would allow the teaching lab to become even more useful as a CSRL resource and as a pedagogical tool.

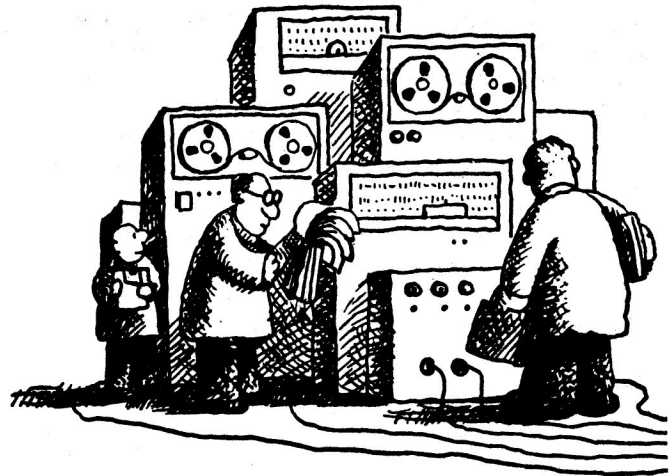
SYSTEM DESCRIPTION

System hardware:

The teaching lab hardware consists of a PDP-8/E with 12K of core memory, a 32K fixed head disk (DF32), eight teletype interfaces, and a connection to the CSRL network.

System Software:

For the teaching lab the three programs necessary for on-line machine language programming have been combined into a single 4K program. This program contains an editor, an assembler, and an interpreter/debugger. Since all users are using the same "copies" of all three programs, user-dependent data must be kept somewhere other than within the program area. For this purpose the remaining 8K of the machine has been divided so as to allocate 1K to each of the eight potential simultaneous users. In each user's area is kept the current "state" of the user in terms of pointers, switches, and other data whose type varies depending on which of the three programs is cur-



rently servicing the user. In order to swap a user in (make him active), it is only necessary to swap the appropriate pointers and switches that define his current state. Only about 100 cells need to be moved, first from the system area to the retiring user's area and then from the activated user's area to the system area, so the transfer occurs rapidly. The disk is used to store the source and binary of the user's program as well as some system files.

Executive:

The "executive" is that part of the teaching lab that determines the next user to be activated and causes him to become active.

Editor:

The editor used in the teaching lab is based on a scrolling, linear data structure, rather than the free-storage buffer, linked-list structure used in DIGITAL's editor. As a result of this data structure, the user is not aware of any auxiliary storage operations and can manipulate his whole edit file regardless of size, a particularly strong advantage in dealing with unsophisticated users.

Assembler:

The teaching lab assembler is based on PAL III, DIGITAL's smallest and simplest (in terms of features available) assembler for the PDP-8. We chose this assembler for inclusion in the teaching lab because of its small size and because it would serve well as an example assembler for CS 802.

Continued on p. 37

SOLO WORKS

A New Name and Some New Ideas From Project Solo

Soloworks is the new name for the continuing NSF-funded Project Solo at the University of Pittsburgh. The project, under the direction of Dr. Thomas Dwyer, is developing some very innovative and unique programs for computer use in secondary schools; a PDP-11/40 RSTS System will be installed shortly to support all aspects of the research effort.

In addition to the many research activities, Soloworks publishes a monthly newsletter that is distributed to one thousand selected educators. Newsletters #23 and #24 are reproduced in part here with the express permission of Dr. Dwyer. From time to time EDU will bring you up to date on the new developments from Soloworks!

SOLOWORKS NEWSLETTER #23 (11/30/73)

As most of our readers know, Project Solo is an experimental program concerned with exploring the potential of computers in the hands of high school teachers and students. The project tested its ideas during 1970-72 in three large public schools in Pittsburgh where several hundred students learned to use computers in "solo" mode. These students (and their teachers) did some very impressive work; their output¹ gave good evidence that the computer is a tool which can help deepen one's understanding of almost any subject.

One of the limitations of computing we discovered during these experiments is the way in which teletype I/O can be an obstacle to "natural" understanding, especially for some of the most interesting parts of mathematics. This is particularly true when one wants to obtain a real feel for the power of mathematics by relating it to science, engineering, the arts, and other disciplines in significant ways.

Some preliminary experiments have convinced us that expanding the world of computer peripherals will make powerful mathematical ideas transparent for lots of students. Developing such peripherals, and coming up with good ideas on how to use them, is the basic concern of the new Soloworks project.²

The name Soloworks is meant to conjure up images of both a place and a philosophy. The place is a lab at the University of Pittsburgh where a small group consisting of project staff, high school teachers, and high school students are working together to develop and test these ideas. This lab is meant to be the prototype of larger math-lab centers to be placed in either conventional high schools, or in central learning centers that serve several high schools.

"Soloworks" is also a statement of philosophy, affirming our belief that any student can be brought into the world of solo-mode learning by an intelligent use of technology. A detailed analysis of what this implies and a game-plan for making it work will be described in Newsletter #26 ("Heuristic Strategies for Using Computers to Enrich Education").

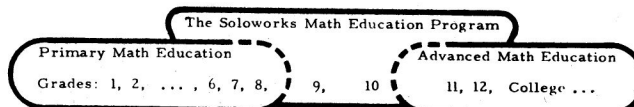
Relation to Math Education

Soloworks is officially a project to develop new approaches to high school mathematics education. Our work is turning out to be quite interdisciplinary, however, drawing upon important ideas from many other fields, particularly computing, science, electronics, engineering, music, and the performing and visual arts. The age level of students with whom we are working falls in the 11-17 bracket, with most of the youngsters in grades 8, 9, and 10.

A simple way to describe where we are in terms of the rest of math education is to say that:

1. We think the first six or seven years³ of primary math education can be seized upon, brought together, and combined with higher math content to really *do* things in Junior-Senior High,
2. After two or three years of doing such important things with math, most students will be in an excellent position to start dealing with (and appreciating) the power of advanced mathematical abstractions.

To say the above in another way, our experience with beginning high school students has convinced us that this is the right time to make mathematics come alive, and that the structure of advanced math, science, and engineering education (grades 11 on up) could become a whole new ball game for youngsters who do important things with math at an early age. We think it is fair to say that our thrust is aimed at a critical transitional period in math education.



The Soloworks Labs

The kind of math to be studied in the Soloworks curriculum is going to be mainly determined by what we call a "top-down" approach. This means that we will start by defining both *research activities* and *skills*; these in turn will determine the mathematics to be studied. This approach results in a very different structure than that obtained by defining content first and then looking for applications. We think the top-down approach will get around the sterility sometimes induced by behavioral objectives that "logically" build on each other in linear fashion.

Our original intent was to develop four lab courses (roughly corresponding to one semester each). A new lab (logical design) seems to be forcing itself on us, giving five courses. The five labs are called Computer Lab, Dynamics Lab, Logical Design Lab, Synthesis Lab, and Modelling/Simulation Lab. Each Lab will focus on the achievement of major skills, and on the pursuit of research projects.

The role of the teacher in such a system is more one of facilitator than lecturer. This is why a technologically based support system is needed. Newsletter #24 will discuss some of the technology we are considering for each lab, and give some examples of the skills and projects involved.

¹ DIGITAL has published four of The Project Solo Curriculum Modules in two volumes: *BASIC Matrix Operations and Computer-Augmented Calculus Topics*.

² The official name of the new project is "A Computer-Based High School Mathematics Laboratory." It is supported in part by NSF Grant EC-38063.

³ These numbers could change radically for children in a primary curriculum of the type being developed by Seymour Papert at MIT.

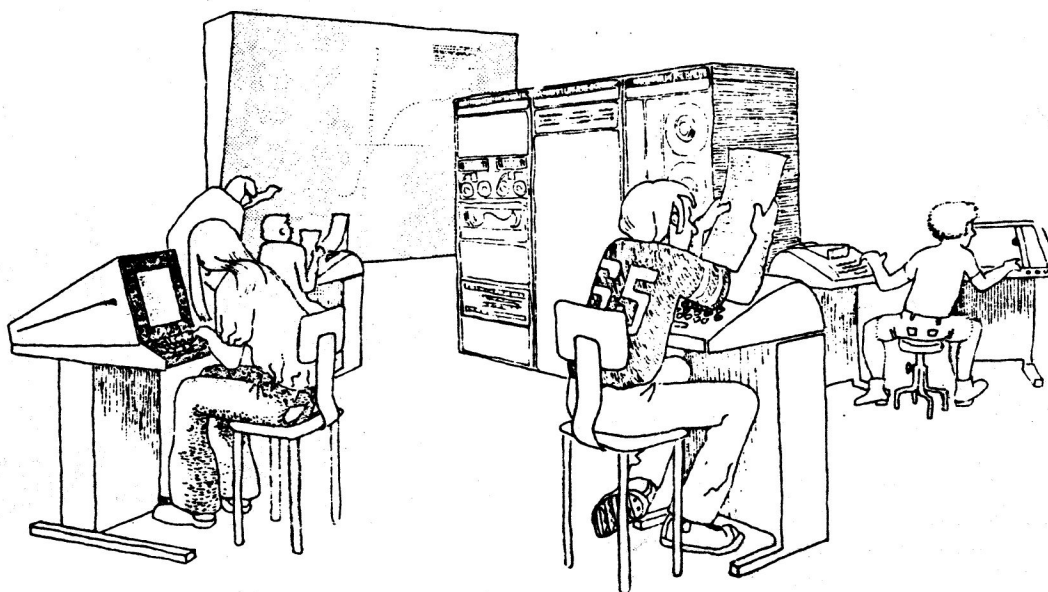
The Computer Lab

The Computer Lab will focus on those aspects of mathematics that are well described by algorithms, and encourage student access to a local computer and terminals. A major skill students will achieve in this lab is that of computer programmer. Examples of some projects that have already been done by students in this lab are developing programs to do accounting and inventory for a small store, programs to play all kinds of games, programs to simulate transportation systems, generation of random ballet dances, a program to interpret "macro" commands for a multi-media show, programs to plot all kinds of mathematical curves and predict and plot their intersections, etc. The significance of the project approach is that although students have available to them a past "heritage" appropriate to attacking the project, they are also expected to develop new and unique extensions of that heritage.

Most of the hardware for the computer lab is "off-the-shelf." A future newsletter will go into some detail about the hardware and software we are using, and the reasons for our choice.

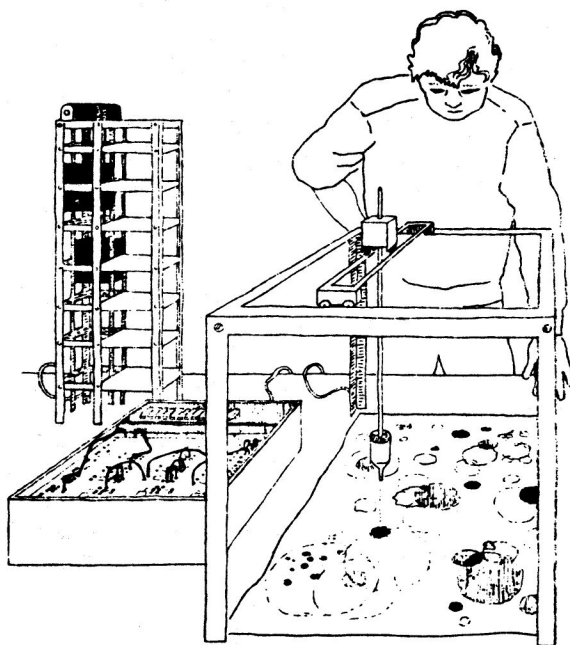
The Dynamics Lab

The Dynamics Lab focuses on mathematics that describes processes that take place in time. One form of technology to be used in this lab is a flight simulator. The skill acquired in using this particular equipment is that of making a full instrument landing, or becoming good at instructing a fellow student to do the same. An example of a project would be to sample analog readings of heading, time, and speed from the flight simulator, translate these into digital data, and then write a program that plots the path of the flight



simulator. Another type of artifact being considered for the Dynamics Lab is the "Rube Goldberg" machine, a gadget designed by the student to do nothing useful, but to be a mind-stretching exercise in imagination. Some other work will center on the use of mechanical ball or disc integrators. We think learning to work with the concept of "rate" can become a very natural thing for relatively young students; rate-of-change and integration are certainly classical examples of powerful mathematical ideas.

In order that other schools can use these same devices with any small computer, it is our intention to transmit data between computer and simulator (or other lab devices) in serial ASCII form. We welcome comments from interested parties on the question of standardization of data formats.



Logical Design Lab

The Logical Design Lab is a new idea which practically forced itself on us as a consequence of wanting to do significant things in the other labs. The technology used will be digital and analog circuit modules. We're tempted to call the skill involved "electronic wizardry." One of the most amazing things about this lab is that one really *does* have to use such things as associative and commutative laws (e.g., to simplify the Boolean equations describing a circuit so that it can be built with a reasonably small number of parts). Projects will include light trees, color organs, cryptography machines, burglar alarms, foolproof control systems for the other projects (e.g., the elevator below), computer-to-lab-device interfaces, etc.

Another area we will include in this lab is suggested by the intense interest in stereo and quadraphonic sound we have noted in young people today. Phrases like "matrix decoder" are known to them, and of great interest. But they don't have the slightest idea how these things work. We intend doing something about that, and math will be center-stage in the explanations.

We would like eventually to come up with some ideas on a kit of logic modules, amplifiers, etc., that takes advantage of recent strides in IC (Integrated Circuit) technology. Such technology is becoming very low cost. We also think the "visible" logic of relays and mechanical linkages should be included.

Synthesis Lab

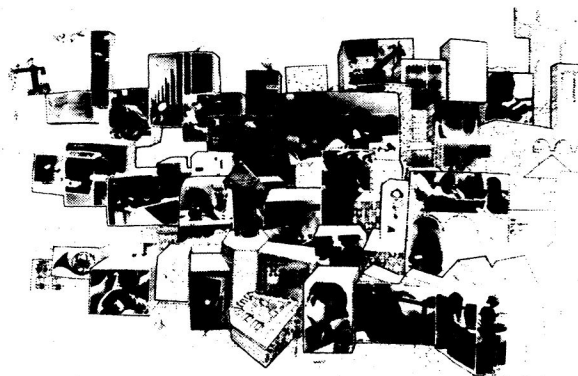
The Synthesis Lab is concerned with mathematics that makes use of the principle of superimposition, producing complex effects by adding together simple ones. Two special pieces of technology we will use are the "Music Monster" (a kind of programmable band-organ), and a multi-media programmer together with suitable projection equipment. The obvious skills associated with these devices are composer and media-designer. Projects will focus on the design, debugging, and performing of original works.

Our initial forays into the music field have been in terms of real organ pipes to be run under program control. Next we'll look at a modularly designed synthesizer. This later technology is replete with application of the concept of function, algebraic products and sums, periodicity, summation of series, local linearity, transforms, etc.

Modeling/Simulation Lab

The Modelling/Simulation Lab uses mathematics as a tool for creating new models of reality that can be studied and manipulated. Some of the models will be physical (e.g., bridges, elevators, lunar landing modules, etc.), some abstract (e.g., an ecology, an economy, etc.). The skill developed here is really that of applied mathematician, while the gamut of possible projects is open-ended. This is because the computer available to students is general purpose, allowing them to simulate systems not heretofore dealt with. Good software is essential too, which is why we are looking into a language like BASIC-PLUS.

The Rube Goldberg machines also fall into this lab (although we're not sure what they simulate!). Perhaps it's semi-accurate to say that they model a fertile imagination. Incidentally, the best model-building sets seem to come from abroad (Automat and Meccano). A really good junk pile seems to be essential.



Soloworks NEWSLETTER #25, perhaps the most innovative to date, is a 30" x 40" 2-Color Poster! The photo-poster illustrates some of the Soloworks laboratories which "are in touch with and building on the heritage of the real world of schools."

SPACEWAR

Continued from p. 31

than with 25 hours of experience on real-time exercises. Of major importance is the concept of analyzing problems for machine language coding.

In order to motivate the student, it was decided that the assignments should involve writing a game that was sufficiently "cute" to make development of the program fun. This program should be modular so that every assignment produces a working program that provides the student with *direct feedback*. The program should also offer sufficient flexibility so that students may exercise creativity in programming.

To accomplish this end the SPACEWAR program was written. SPACEWAR has two spaceships operating under the influence of gravity from a star; the ships obey all of Newton's Laws (approximately) and fire missiles which also move under a gravitational field. Many options (implemented under interrupt control) are available to demonstrate the use of interrupt service routines and I/O overlap. These options include changing the strength of gravity, the types of ships displayed, thrust constant rockets, conservation of rotational momentum of the ships, and reflection at the scope screen edge. An added advantage of these options is that they encourage the student to be creative in writing his own version of the game.

In order to give positive feedback, subroutine modules and a working program are distributed to students at the completion of each programming task; students learn a great deal from studying these subroutines. For example, one of the first assignments is to display a spaceship using a supplied display subroutine. Next, the ship is rotated, and students are given the routine to display a ship. At every stage, one of the supplied routines performs what the students have just accomplished. This approach is extremely useful for comparative purposes, for an excellent way to learn computer programming techniques is by analyzing a similar program.

Our experience has shown that it is almost impossible to teach a student machine language programming techniques in a traditional manner. Instead, the instructor must motivate the student to teach himself, supplying guidance and help along the way. This approach has been particularly successful in our assembly language programming course, bringing students rapidly to the point where more traditional teaching methods may be used.

Stanford University music students are interacting with a DECsystem-10 through a terminal to learn concepts of music. The computer drives an organ as a sound source, and the student answers, identifying notes and phrases. The system was developed by Wolfgang Kuhn, professor of music and education.

LABORATORY

Continued from p. 33

However, the teaching lab design permitted changes to this basic approach of PAL III. First, input comes from the internal data structure of the editor, rather than paper tape.

Second, the editor is co-resident in core with the assembler, allowing the assembler to use the editor support routines to fetch characters and do listings.

Third, the symbol table is not maintained within the assembler but in the individual user areas. This prevents one user from destroying everyone else's programs if he should happen to redefine a permanent symbol and permits a larger user symbol table than could otherwise be implemented.

Fourth, the binary output goes to a working storage area on disk, allowing the listing and the error messages to be put on the teletype during pass two.

Interpreter/debugger:

The interpreter/debugger is a new program, not an adaption of a previously existing program. It is included as part of the teaching lab system in order to satisfy three demands made on the system by users in the process of developing correctly operating programs.

The first is the "safe" execution of an untried and probably incorrect program in a multiprogramming environment that has no hardware safeguards to protect the system or other users from an incorrectly operating program. As a result, user programs are interpreted rather than executed directly.

The second demand is for the ability to modify programs online (batch facilities) and to exercise program segments without incurring the time penalty of editing and reassembling. The result is the inclusion of the ODT-like features.

The third demand is the ability to provide information on program performance and operation up to the point of a known erroneous operation. Here the result was to include SNAPS (similar to selective, dynamic core dumps) and a history table which can be examined by the user.

WHAT NEXT?

This article has described the teaching lab as a whole and its subsystems in terms of their own operations, their interactions and their demands on the system. In order to study user behavior and consequent system behavior, we have designed and implemented a data gathering and data analysis system to be used with the teaching lab. Our research has enabled us to define design objectives for a new system, simple enough for the novice programmer, able to support at least 12 users simultaneously, and capable of supporting the hardware necessary for a hardware interfacing laboratory.

SOURCE MATERIAL FOR EDUCATORS



THE JOURNAL

THE Journal is a new educational periodical, designed to serve as a medium of exchange for educators and industry representatives. Committed to the field of educational technology, *THE Journal* provides a wide range of timely reports and articles written by experts in the field.

THE Journal's philosophy is best stated in the words of the editor, John F. Martell:

THE Journal's editorial goal is to provide significant source data to those members of the educational community seeking to employ new technological systems in education and to serve as a meeting place for those educators and the producers of technical products.

Our purpose then is to provide a forum for the free interchange of information needed by both the educational and industrial communities. . . .

Industry needs guidance from the educators and the educators need evidence of available technology that will materially assist them in achieving the increased performance that all are seeking.

To obtain a subscription to *THE Journal* send \$15.00 (1 year) to *THE Journal*, P. O. Box 992, Acton, MA. 01720. And mention EDU!

COMPUTER SCIENCE MATERIAL

DIGITAL, over the years, has published a wide range of curriculum material suitable for the study of computer science. Summarized here are the text and resources appropriate for computer familiarization courses and for the study of computer programming and hardware familiarization. All material can be ordered directly from the Software Distribution Center, Bldg. 1-2, DIGITAL.

COMPUTER FAMILIARIZATION MATERIALS

<i>Getting Started in Classroom Computing</i>	\$ 1.50
<i>101 BASIC Computer Games</i>	5.00
<i>Understanding Math and Logic Using BASIC Computer Games</i>	1.50
<i>Population: A Self-Teaching BASIC Primer</i>	2.00
<i>EduSystem Handbook</i>	5.00
<i>23 Slides For Teaching BASIC (plus Teacher's Guide)</i>	20.00
<i>Huntington II Project Filmstrip and Cassette</i>	15.00
<i>Problems For Computer Mathematics</i>	1.25
<i>Huntington II Simulation Program: POLUT (Student, Teacher, Resource Handbook and Paper tape)</i>	2.10

COMPUTER PROGRAMMING AND HARDWARE FAMILIARIZATION TEXTS

<i>Introduction to Programming</i>	\$ 2.00
<i>EduSystem Handbook</i>	5.00
<i>OS-8 Handbook</i>	9.95
<i>OS-8 Mark Sense Batch User's Guide</i>	7.50
<i>BASIC-PLUS Language Manual</i>	7.50
<i>RSTS System User's Guide</i>	7.00
<i>BASIC/RT-11 Language Reference Manual</i>	7.50
<i>RT-11 System Reference Manual</i>	7.50
<i>BASIC/PTS User's Manual</i>	12.50
<i>23 Slides For Teaching BASIC (plus Teacher's Guide)</i>	20.00
<i>Small Computer Handbook</i>	1.00
<i>Logic Handbook</i>	1.00
<i>Computer Lab Texts</i>	
Student Workbook	1.00
Teacher's Guide	5.00
<i>Experiments in Logic Design and Computer Interfacing</i>	3.00
<i>Introduction to Data Communication</i>	1.00
<i>Logic System Design Handbook</i>	1.00

Detailed descriptions of these items may be found in the Curriculum Material Product Catalog, available from Communication Services, PK1, DIGITAL.

BOOK REVIEWS

A Collection of Programming Problems and Techniques by H. A. Mauer and M. R. Williams. Prentice-Hall, Inc. Englewood Cliffs, N.J., 1972. \$6.95

The problems contained in this book were originally conceived by the authors for use in their own computer science course at the University of Calgary. Despite the proliferation of texts for programming languages and techniques, the authors realized that a supplemental text with a comprehensive collection of problems for computer solution did not exist.

A Collection of Programming Problems and Techniques fills the void expertly; it contains a wide variety of problems suitable for computer solution. Many of the problems require only a high-school level knowledge of mathematics. Problems are drawn from topics including number theory, games, equations, random numbers, sorting, statistics and plotting. All problems are stated in English sentences and are not related to a specific computer language. Mathematical rather than computer solutions are contained in the appendix of the volume.

Designing Classroom Simulations by Glenn S. Pate and Hugh A. Parker, Jr. Fearon Publishers. Belmont, Cal., 1973.

Designing Classroom Simulations is a 70-page pamphlet that provides guidelines and ideas for designing original or adapting existing classroom simulations. Although the booklet does not deal specifically with computer simulations, many of the principles relate directly to the design and evaluation of computer simulations as well.

Chapters include topics such as selecting objectives; ensuring student involvement; designing the framework; using, evaluating and improving simulations. Suggested topics appropriate for classroom simulations are also included.

Electric Media by Les Brown and Sema Marks. Harcourt Brace Jovanovich, Inc. New York, N.Y., 1974.

Electric Media is a highly contemporary treatment of society's "love-hate relationship" with the two most controversial electric media of our age—television and the computer. Based on the fact that the public both marvels at and is intimidated by computers, the second section of this book provides numerous views of the many facets of daily life that are affected by computing technology.

Sema Marks, author of the computer segment of the volume, presents the fundamentals of the "remarkable," "ubiquitous," "intelligent" and "creative" computer in an unquestionably unique and creative manner; her insight and originality make this volume an ideal resource for computer familiarization and computer science coursework. In fact, it's a perfect just-for-fun book for computer novices and experts, too!



Clarifying Values Through Subject Matter by Merrill Marmin, Howard Kirschenbaum, and Sidney B. Simon. Winston Press, Inc. Minneapolis, Minn., 1973. \$2.25

Clarifying Values Through Subject Matter is designed to make subject matter more relevant and exciting through classroom value clarification activities. Three levels of teaching (facts, concepts, and values) are fully described. The book offers strategies, ideas for discussion, and guidelines for designing a value unit in all subject areas.

Applications of Computer Systems compiled by Richard A. Bassler and Edward O. Joslin. College Readings, Inc. Arlington, Va., 1974. \$3.95

This text examines fundamental computer concepts and computer applications through a series of selected articles and readings. Some 40 articles discuss the applications of computers in business and finance, education, transportation, home economics, medicine, manufacturing, engineering and architecture. The collection makes ideal supportive material for computer familiarization and computer science curricula.

The Terminal Man by Michael Crichton. Alfred Knopf, Inc. New York, 1972. \$6.95

This novel combines authentic description with hair-raising suspense to open up for the reader a new area of modern science: surgical-computer mind control.

Psychosurgery is performed on a violent paranoid who has twice attempted to kill. A team of surgeons conducts a delicate operation, connecting 40 wires from the patient's brain to a microminiature computer implanted in his neck. It is the job of the computer to detect the start of a violent seizure and prevent it by stimulating a pleasure or calm node of the brain. The tension builds throughout the book from the initial conflict between the doctors to the final terrifying results when the patient escapes from the hospital before the computer program is tested.

Psychosurgery of the kind Crichton describes is already taking place in medical research centers today, thus making mind control a key scientific and moral issue of our time. Crichton takes it out of the realm of the abstract, and makes immediate its workings, its dangers, and its implications in a novel that provides urgent information and superb entertainment.

NEW FROM HUNTINGTON II

The science teachers in our reading audience will be glad to know that the following three new additions to the Huntington II Simulation Program collection are now available:

HARDY

Subject: Biology

Level: Grades 10-14

Description: HARDY simulates the Hardy-Weinberg Principle of population genetics and assists the student in determining the proportion of a given population that are homozygous dominant, homozygous recessive and heterozygous dominant with regard to a specific trait.

PH

Subject: Biology

Level: Grades 10-14

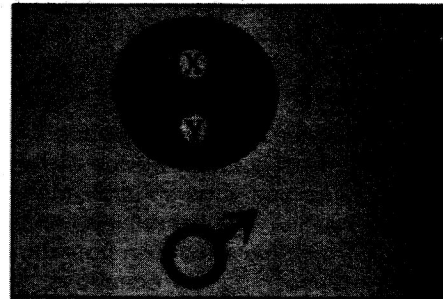
Description: PH consists of three laboratory investigations dealing with the specificity of enzymes.

SCATR

Subject: Physics

Level: Grades 11-14

Description: SCATR simulates alpha particle scattering as demonstrated in the laboratory and in accordance with three theoretical models of the atom: the hard sphere, the Thomson and the Rutherford or nuclear model. This simulation provides empirical evidence of the validity of the nuclear model of the atom while it permits students to study the mechanism of scattering on the level of individual atoms.



PRICE SCHEDULE:

	HARDY	PH	SCATR
Student Workbook	.50	.50	.50
Teacher's Guide	.50	.50	.50
Resource Handbook	1.00	1.00	1.00
Paper Tape(s)	1.00	1.00	3.00
Individual Packet	3.00	3.00	5.00
Classroom Packet	23.00	23.00	25.00

For order forms and additional information regarding Huntington simulation materials, obtain a copy of the Curriculum Material Product Catalog from Communication Services, PK1, DIGITAL. All orders for Huntington materials should be forwarded to the Software Distribution Center, Bldg. 1-2, DIGITAL.

GETTING STARTED?

Just getting started in instructional computing? Well, DIGITAL's newest publication was written to help students and faculty meet the computer!

Getting Started in Classroom Computing is a little booklet that contains six lessons that can be used with or without a computer. Faculty and students interact in groups to solve the game problems on paper or through role playing—then the same games can be played using the computer.

Getting Started In Classroom Computing is a companion volume to *101 BASIC Computer Games* and is ideal for computer familiarization courses and faculty seminars.

Obtain a copy by sending \$1.50 (add \$5.00 for *101 BASIC Computer Games*) and \$.50 postage and handling to the Software Distribution Center, Bldg. 1-2, DIGITAL.

W. E. S. BULLETIN

W.E.S. Bulletin is a three year old newsletter published ten times per school year that caters to educational leaders who desire to see responsible changes made in American education. The newsletter is sold on a subscription basis to school superintendents, curriculum consultants, teacher consultants, principals, department heads and teachers.

Each issue contains articles on open campus, open concept, the ungraded classroom, individualized instruction, team teaching, humane schools and the use of aides and paraprofessionals. Any educational development that represents the successful implementation of new techniques can be found within the covers of the *W.E.S. Bulletin*.

The subscription rate for one year is \$5.00. To order the *W.E.S. Bulletin* or to obtain a sample copy of the newsletter write to Watman Educational Services, Box 457, Henniker, N.H. 03242. Tell them we sent you!

NEW CURRICULUM PACKAGES

Two new curriculum packages are now available from the Software Distribution Center. Both are intended to support the integration of computer studies into existing curricula; the packages contain a wealth of applicable material published by DIGITAL as well as newly published curriculum guides and supportive material. Both packages may be obtained from the Software Distribution Center, Bldg. 1-2, DIGITAL.

BASIC PROGRAMMING CURRICULUM PACKAGE

Contents:

Population: A Self-Teaching BASIC Primer;
DIGITAL

EduSystem Handbook; DIGITAL

101 BASIC Computer Games edited by D. H. Ahl;
DIGITAL

23 Slides For Teaching BASIC (with teacher's
guide); DIGITAL

*Bibliography of Texts and Resources on the Uses
of Computers in Education;* DIGITAL

A Short Course in BASIC by P. H. Ellsworth;
DIGITAL

A Curriculum Guide For Teaching BASIC by S. R.
Bower; DIGITAL

Price: \$50.00

Order No.: QFE02-PZ

COMPUTER MATHEMATICS CURRICULUM PACKAGE

Contents:

BASIC Application Programs—Mathematics I;
DIGITAL

BASIC Application Programs—Mathematics II;
DIGITAL

BASIC Application Programs—Plotting; DIGITAL

Huntington I Application Programs—Mathematics;
DIGITAL

BASIC Matrix Operations, Project Solo; DIGITAL

*Computer-Augmented Calculus Topics, Project
Solo;* DIGITAL

101 BASIC Computer Games edited by D. H. Ahl;
DIGITAL

*Understanding Math and Logic Using BASIC Com-
puter Games* by D. H. Ahl; DIGITAL

Getting Started in Classroom Computing by D. H.
Ahl; DIGITAL

Problems For Computer Mathematics by Ronald
Allison; DIGITAL

Advanced Problems For Computer Mathematics
by Robert Albrecht; DIGITAL

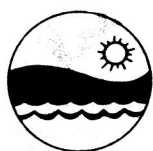
*Bibliography of Texts and Resources on the Uses
of Computer in Education;* DIGITAL

*A Curriculum Guide For a School Computer Pro-
gram in Mathematics* by G. Albert Higgins, Jr.;
DIGITAL

Price: \$25.00

Order No.: QFE03-PZ

For additional information regarding DIGITAL's wide assortment of supportive classroom material, obtain a copy of the Curriculum Material Product Catalog from Communication Services, PK1, DIGITAL.



ENVIRONMENTAL EDUCATION MATERIAL

The Environmental Education Material Catalog, published by the National Science Teachers Association, contains a collection of reviewed and evaluated free and low-cost materials for science, social studies, and industrial arts. The catalog is designed to help teachers and students obtain information regarding society's environmental problems and to contribute to wise decisions regarding solutions to these problems. References are appropriate for early childhood through secondary education.

To obtain a copy of the catalog, send \$2.00 to National Science Teachers Association, 1201 Sixteenth Street, NW, Washington, D.C. 20036. And mention EDU!

ECOSOURCES

Ecosources is a monthly publication that contains references and bibliographies of classroom materials for a variety of subject areas. Each month, *Ecosources* features a different subject area; many of the subjects may be adapted to computer use or used as originally intended.

To date, some twenty-four issues have been published. Selected titles include:

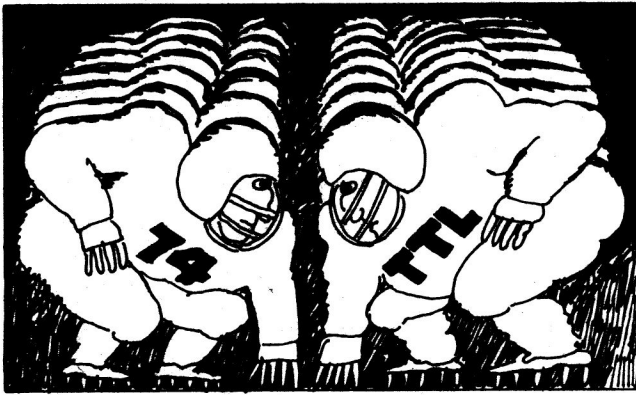
No. 1 Awareness

No. 2 Environmental Problems of Energy

No. 9 Population

No. 12 Environmental Games and Simulations

A summary of all back issues are contained in the latest issue, which can be obtained at no charge by sending a stamped, self-addressed envelope to Ms. Jane Woerner, Freeland H.S., 710 Powley Drive, Freeland, Michigan 48623. One envelope must be forwarded for each copy of *Ecosources* ordered. Please don't forget to mention—EDU!



FOOTBALL SCOUTING

How did the Cabrillo (CA) High School football coaching staff save between eighteen and thirty hours of coaching time per week last season? They used Hal Singer's football scouting report system and an EduSystem 30—that's how!

Hal is the director of the Cabrillo Computer Center and author of the football report system. Before the analysis programs were written, coaches from Lompoc and Cabrillo High Schools used the traditional flip-card notebook for recording plays in the three to five games scouted weekly. That information was then punched into wire-sort cards and analysis performed by manual wire sorting; a scouting report and game plan book were produced.

Using the football scouting report programs, the results of each play are coded on special mark sense cards, and the game is exhaustively analyzed by Cabrillo's EduSystem 30. The programs are written in EduSystem 30 BASIC and print the following comprehensive reports:

- formation information
- passing information
- field position information and down-distance tendencies
- hash mark tendencies
- big plays, most frequent plays and play sequence

The Lompoc System provides slightly different printed reports containing the following information:

- running formation information
- pass formation information
- passing area information
- field position tendencies—running and field position tendencies—passing
- down-distance tendencies and hash mark tendencies
- big plays, most frequent plays and play sequence

Full documentation of the Cabrillo and Lompoc football report system programs maybe obtained at no charge from the DECUS Librarian, PK3-2, DIGITAL (Order No. BASIC 8-34). Both programs may be easily modified to suit the requirements of the coaching staffs at individual schools.

COMPUTER TUTOR

Like the idea of a computer tutor for learning BASIC? Well, if you do, DECUS has just the thing for you—seventeen tutoring programs, written in BASIC-PLUS for PDP-11 RSTS and RSTS/E Systems.

TUTR01 through TUTR17 are self-documented and contain complete instructions, so that no textbook or learning guide is necessary. All that's needed is a RSTS-11 system and some extra time!

Documentation (no charge) and DECtape containing the seventeen BASIC tutoring programs (\$17) may be ordered directly from the DECUS Librarian, PK3-2, DIGITAL. Order No. RSTS11-22.



CLOZE READABILITY

Dr. Susan Gross and E. Joseph Kocinski of the Matteson (Illinois) School District 162 have developed under a Title III grant a unique computer program for determining readability. A readability formula is used to determine if a student has achieved an appropriate reading level to master specified textual material. In many schools, readability formulas are being used effectively to match student ability and textual material for individualized instructional programs.

The computer program is based on the Cloze Readability Formula—one of several available formulas. The Cloze Readability Formula is the most popular for determining the readability of reading, social studies and science texts for grades 5 to college level. This formula is based on a table-lookup system whose parameters are average letters per word and average letters per sentence.

The program, written in EduSystem 25 BASIC, accepts a 250 word (minimum) passage. Output produced includes the average number of letters per sentence and the reading comprehension level necessary on the Stanford Achievement Test to understand the material.

For additional information contact the authors at Matteson School District #162, Matteson, Illinois 60443.

GAMES FOR COMPUTER SCIENCE

Many commercially available board games and devices designed for classroom use are ideal for integration into computer science and familiarization coursework. The non-computer games listed below may be easily adapted for use in these curricular areas—as an introductory programming activity or as an exercise to develop logical and analytical abilities.

ACQUIRE: Simulates the stock market

2-6 players

Grades 9 and above

3M Company, St. Paul, Minnesota

BASIC STRIPS KIT: Forty-five packets of cardboard strips that assemble into workable computer programs

Grades 7 and above

TTT Math Lab, Patrick Henry Junior High School, Granada Hills, California

BINARY: Deals with binary arithmetic

2-4 players

Grades 7 and above

Product Design and Development, Cocoa Beach, Florida

CARDIAC: A cardboard computer for teaching fundamentals of computer operation. Teacher's guide included.

Grades 7 and above

Bell Telephone Laboratories, Murray Hill, N.J.

COMPUTER LAB: Introduces hands-on wiring of logic modules. Workbook and teacher's guide included.

Grades 9 and above

Digital Equipment Corporation, Maynard, Mass. 01754

HI SPOT: An inverse domino game involving memory and strategy

Grades 7 and above

Pressman Toy Corporation, Brooklyn, New York

JUMPIN: Strategy game involving twelve pawns

2 players

Grades 7 and above

3M Company, St. Paul, Minnesota

KALAH: An ancient game of counting

2 players

Grades 3 and above

Creative Publications, Palo Alto, California



MINICOM

MINCOM is a BASIC program which simulates a 64K word computer on a DECsystem-10. Designed for and used in the computer science curriculum at Hatfield Polytechnic, England, *MINCOM* demonstrates the following machine features:

- (1) Use of a computer console.
- (2) An instruction set with memory-reference and addressless instructions (including instructions for two's complement arithmetic, logic, skips, shifting, I/O, etc).
- (3) Use of flags and indicators for overflow.
- (4) Sophisticated hardware features: indirect addressing, immediate operands.
- (5) System software: binary loaders, simple assemblers, core dump routines.

Since the introduction of *MINCOM* into the curriculum at Hatfield, a *MINCOM* hardware terminal has been interfaced to the DECsystem-10 to separate the functions of *MINCOM* from the simulated operation of the actual console terminal.

More information regarding *MINCOM* can be obtained by writing to Mel Jackson and Brian Smith, Hertfordshire Advisory Unit, Hatfield Polytechnic, Hertfordshire, England. Mention EDU!!



PROJECT CLUE REPORT

An extensive review of the instructional use of computers was pursued by the Center for Research on Learning and Teaching at the University of Michigan during 1969 and 1970. The final report of that study, *An Evaluation Review of Uses of Computers in Instruction* (Project CLUE), has been reprinted for the second time. Although some of the items are dated, much of the information and advice continues to be useful in planning and review of the instructional use of computers.

The report includes guidelines to information sources, guidelines for current uses, and detailed sections on management and curriculum development.

To order the 500-page report send \$5.00 (checks payable to the University of Michigan) to: Karl L. Zinn, Project EXTEND, 109 E. Madison, Ann Arbor, Michigan 48104.

In the spirit of information exchange, the report may be obtained for \$2.00 and an article or report on instructional computing on the campus of the interested educator. Alternatively, \$2.00 and a commitment to provide a review or a comment on the Project CLUE report or a similar publication will also earn a copy of the report.

ANNUAL BIBLIOGRAPHY

Every January issue of the *Computing Newsletter For Instructors of Data Processing* contains a bibliography of texts and resources for a wide range of computer-related subject areas. The seventh edition of the bibliography contains 17 pages of references for students, teachers and data processing managers; classifications include introduction to E.D.P., computer programming, systems analysis and design, computer-based management information systems, computer applications, computer mathematics, familiarization and simulation.

A copy of the January 1974 bibliography can be obtained only through subscribed membership. A subscription buys 9 issues (1 year) for \$11.00 and may be ordered from J. Daniel Couger, Editor, *Computing Newsletter*, P. O. Box 7345, Colorado Springs, Colorado 80933. Mention EDU!

THE MATH GROUP

Remember the article entitled "Computer Conversations" in EDU #11? The Math Group is the organization that produces the Computer Conversations cards for teaching BASIC. And they've extended an offer to our readers! A free set of 41 cards and a teacher's solution guide (a \$7.50 value!) will be sent to any educator who submits an original problem for computer solution that the Math Group publishes in a future set of cards! In the case of duplicate entries, only the first submission will be awarded the complimentary Computer Conversations Package.

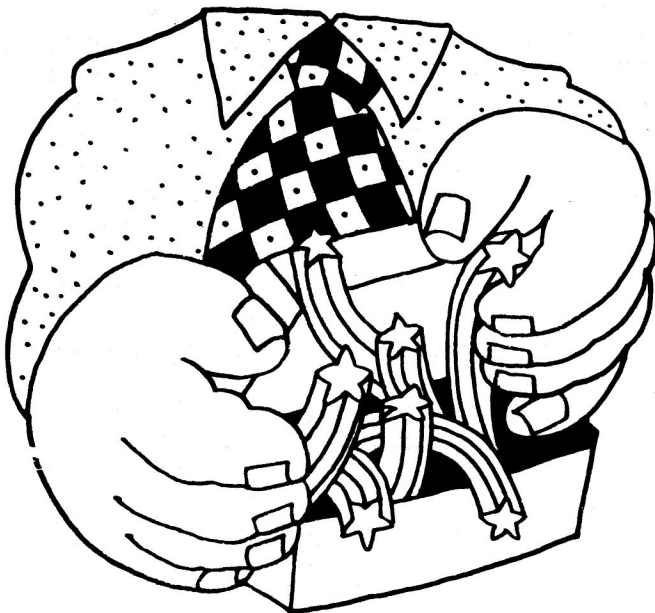
Send your submissions to The Math Group, c/o Earl Orf, 5625 Gerard Avenue South, Minneapolis, Minnesota 55419. Mention EDU!

BCS YEARBOOK

The 1972-73 British Computer Society (BCS) Educational Yearbook is being distributed in the Americas by ACM as a service to the members of the Special Interest Group on Computer Uses in Education (SIGCUE). The theme of the yearbook is the educational activities of the International Federation for Information Processing (IFIP), and the yearbook contains a bibliography of all internationally published material in the field of computer education. An impressive collection of books, films and computer courses, academic and industry-sponsored, as well as editorials, articles and papers are included in this 206-page volume.

The BCS Yearbook may be ordered for \$10.00 (checks payable to SIGCUE Exchange—payment must be enclosed) from Karl Zinn, Center for Research on Learning and Teaching, University of Michigan, 109 East Madison Street, Ann Arbor, Michigan 48104. Mention EDU!

IDEAS AT WORK



PROBLEM CORNER

This deceptively difficult problem first appeared in the BASIC Information for Teachers (B.I.T.) Newsletter, December 1973. We're sure it will challenge your most advanced and creative students!

Write a BASIC program that will print all permutations of N things taken N at a time for all $N \leq 10$. A sample run of your program might appear as:

```
RUN
? 3
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
READY
```

MIDNIGHT TO MIDNIGHT

We've even had trouble because they were staying after school was closed . . . They would hide . . . until everybody disappeared and then they would come here (computer room) . . . They found them at 8:00 at night sometimes . . . They want to be on these machines from midnight to midnight. We have to throw them out at night. It's really fantastic.

—Anonymous teacher from a Project Solo school,
Pittsburgh, Pa.

COMPUTER OLYMPICS

Classroom Olympics is one way to teach students the importance of accuracy and precision in measurement, while it also gives the class an opportunity to utilize the lightning speed of the computer for calculations!

Groups of 4 or 5 students work together to invent an event; use a plastic golf ball for the shot put, a straw for the javelin throw, a paper plate for the discus and measure an area for the broad jump. Each group can take measurements during the Olympics, and use the computer to calculate the results. The Olympic "gold medalists" get free computer time!



KEYS TO SUCCESS

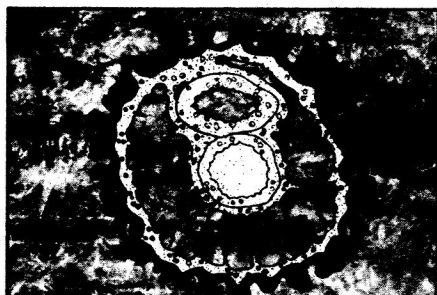
Paul Shapiro, Mathematics Department Chairman at Newton (Mass.) High School, knows about the keys to successful implementation of a computer instructional program. One is *communication*. Another is *experience*. And Paul is doing his part to ensure that Newton's RSTS-11 installation is successful.

CONNECTS/11 is the user newsletter edited and published by Mr. Shapiro; it is designed to serve as a communication medium for the faculty of Newton North and South High Schools.

Because experience is such an important ingredient in ensuring wide faculty involvement, the faculty at Newton sponsors workshops for all interested teachers and administrators. Last winter, two three-session seminars on social studies and science simulations were held at both high schools.

A GAME OF LIFE

The following is an excerpt from the new DIGITAL volume entitled Understanding Mathematics and Logic Using BASIC Computer Games, by David H. Ahl. This treatment of The Game of "Life" is contained in a chapter on Abstract Models; much of the discussion and the exercises were designed by Paul Shapiro, teacher at Newton High School, Newton, Mass., for use as an open-ended project in his untracked computer course. The entire project description with attendant exercises is reprinted here. The Game of "Life" is an excellent example of a problem which can be extended to all educational levels (from secondary to college) and is, at the same time, a valuable exercise for students of different ability levels. The BASIC program of The Game of "Life" can be found in the 101 BASIC Computer Games book.



Introduction

Generally, a mathematical model is a representation of some real-life process, expressed in mathematical form (such as a set of related equations) or in algorithmic form (such as a computer program). Usually the model is by necessity a simplification of the actual process, since real-life processes tend to be highly complex. One advantage of embodying the model as a computer program is that we can run the program and thus simulate the process being modelled. By varying certain features of the program, we can learn something about the relationships between the components and the overall structure of the process. In addition, if the output does not sufficiently coincide with observed reality, the model can be revised and improved.

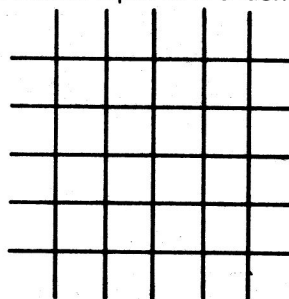
It is also possible to model a purely abstract process. We don't often see this done. After all, if someone asked you to describe some abstract process, what would you say? However, many games start out as purely abstract processes. For example, tic-tac-toe or checkers are abstract from the point of view that they represent no real-life process. Occasionally, it turns out that an abstract process represents a real-life process either by accident or design. The model described below is one such abstract game that in some ways actually represents life itself.

The Game of "Life"

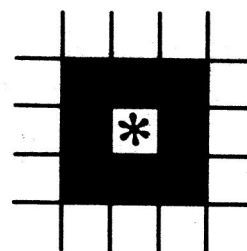
The Game of "Life" was devised by John Conway, a mathematician at the University of Cambridge, and made popular by a series of articles written by Martin Gardner in recent issues of *Scientific American*. Ever since the first article appeared in October 1970, hundreds of mathematicians throughout the world have become fascinated with the model and have been exploring its properties.

The game consists of following the successive generations of a particular imaginary type of cellular life-form. The life processes of these cells are represented by the following mathematical model, which captures several properties common to all life-forms.

- (1) **World** — Cells live on an infinite two-dimensional plane of squares (like an infinite checker-board except that all squares are identical).



- (2) **Neighborhood** — Each square has eight "neighbor" squares. In the diagram below, the neighbor squares for the square with the asterisk (*) have been colored in.

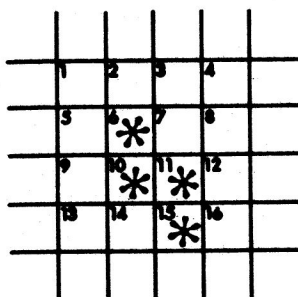


- (3) **Survival** — A cell (always represented by an *) which is living in generation n , will remain living in generation $n+1$ if and only if it has exactly 2 or 3 living neighbors in generation n .
- (4) **Death** — However, in all other cases the cell dies. Specifically: If it has 0 or 1 neighbors it dies from isolation. If it has 4, 5, 6, 7 or 8 neighbors, it dies from overpopulation.
- (5) **Birth** — If a square is empty during generation n , a living cell will be born into that square during generation $n+1$ if and only if that square had exactly 3 (no more, no less) living neighbors during generation n .

The only trick is to remember that all survivals, deaths, and births occur *simultaneously*, and so the simplest way to keep the "bookkeeping" straight is to have two separate copies of the world—one for the old generation and one for the new one you are forming. For each square in the old world, decide what its state will be next time, and mark this down *in the corresponding square* in the new world.

The game is played simply by picking some initial starting pattern and watching the development of some very interesting, and often beautiful, patterns of symmetry. However, the player must be extremely careful because mistakes are easy to make.

As an example, we will trace three generations of the following initial pattern (we have numbered some rows and columns for reference purposes only):



Following the rules of our model:

No births will occur in squares 1, 2, 3, 4 or 5 because none has *three* living neighbors

The cell in square 6 will survive because it has two living neighbors (10 and 11)

A birth occurs in square 7 because there are three living neighbors (6, 10 and 11)

No birth occurs in squares 8 or 9

The cell in square 10 survives because it has three living neighbors (6, 11 and 15)

The cell in square 11 survives also because it has three living neighbors (6, 10 and 15)

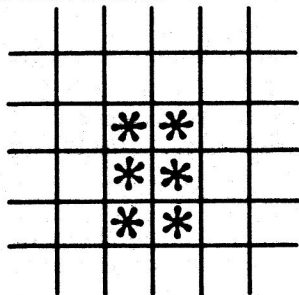
No birth occurs in squares 12 or 13

A birth occurs in square 14 because there are three living neighbors (10, 11 and 15)

The cell in square 15 survives because it has two living neighbors (10 and 11)

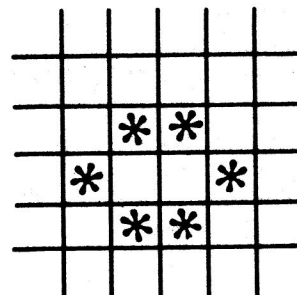
No birth occurs in square 16 because it only has two living neighbors

During this process, we have been filling in a picture of G1, and the end result is:



EXERCISE - 1

Using pencil and paper, carefully compute G2, the next generation for this same society of cells. If you do it correctly, you will find that G2 is the pattern which Conway calls the "beehive":

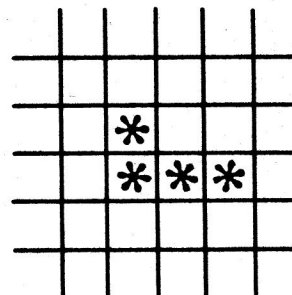


EXERCISE - 2

Now compute G3. If you are again careful, you will discover that G3 is identical to G2. Why does Conway call the beehive a "still-life"? If you are not sure, think about G4, G5, G6. . .

EXERCISE - 3

Using pencil and paper, compute G0, G1, G2 and G3 for the initial pattern below:



If you do it correctly, G3 should look familiar to you.

PROJECT - 1

By now you've no doubt noticed that with pencil and paper, this game is an extremely slow process, and mistakes are all too common. If we ever hope to look at more than a few patterns, we're going to have to turn to the computer for help.

Write a computer program which simulates "Life" for *any* given initial pattern, and which has the following features:

- (1) Allow for as large a world size as your particular computer facility will permit (obviously an infinite plane is not possible in a finite memory). You will probably want to use array structures with two subscripts (row and column).
- (2) Whatever world-size you are limited to, make sure your program doesn't try to allow births outside your world, even though properly these *would* occur on an infinite plane.

- (3) Make sure your algorithm allows all survivals, deaths, and births during a given generation to occur *simultaneously*, as discussed above.
- (4) Allow the user to input the initial pattern in a convenient format, such as pairs of (row, column) coordinates.
- (5) Make your program efficient and your output as close to the format of the pictures above as possible.

Once your algorithm is designed, and your program is written, debug your program by running it on the following initial G0 pattern, and carefully check your output vs. the results below:

G0	G1	G2	G3	G4	G5
*	***	* *	*	***	* *
***	***	***	* *	* *	* *
	*		*	***	*
			*		*

etc.

Warning!

Depending upon the world-size you are limited to, certain "large" patterns may grow differently than they would on an infinite plane.

If the society of cells above, however, fits inside your world-size, you will notice an interesting cyclic pattern beginning at G0, which Conway calls "traffic lights".

PROJECT - 2

When your program is thoroughly debugged and operational, or using the existing LIFE program, the real fun comes in thinking up initial patterns and watching them grow. Interesting situations to watch for are:

- (1) Other "still-life" societies (like the "bee-hive")
- (2) Other "cyclic" societies (like the "traffic lights")
- (3) A society which lives for an extended period of time without dying, becoming still, or cycling.

Bring your findings to class for comparison and discussion.

OPTIONAL PROJECT - 3

Find copies of the October 1970 and/or February 1971 issues of *Scientific American* and read Gardner's articles on "Life." You may want to run your program on some of the societies he describes, such as: diagonal chains, the R pentomino, the latin cross, the cheshire cat, and many others. Your school library may subscribe to the magazine. If not, the public library near you will certainly have it.

OPTIONAL PROJECT - 4

Try to think up changes in the model (and your computer program) which will drastically alter the life patterns of the cells, i.e., by modifying the rules for birth or death or both. Based upon your experience so far, try to come up with sets of rules which will lead to more populous societies, or more sparse societies, or societies which are less symmetric than those of "Life," etc. The range of possibilities is very large. Bring your findings to class for comparison and discussion.

OPTIONAL PROJECT - 5

Make some major modifications in your computer program to make it more *general*, by allowing the user to specify the particular model he wants to investigate. For example, you might have your program begin by posing the following questions to the user:

How many neighbors for survival?

How many neighbors for birth?

Then, if the user answered 2, 3 for the first question and 3 for the second, your program would follow the rules of "Life." But if he gave other answers, the program would simulate for him some other model he wants to investigate.

OPTIONAL PROJECT - 6

Is there any way you can streamline your program so that you can enlarge the size of the world it currently handles?

OPTIONAL PROJECT - 7

How might you alter the general concept of "neighborhood" so that entirely different models could be tested? How would your computer program have to be changed in order to simulate these new models?

OPTIONAL PROJECT - 8

It is possible to place two different types of life forms in the world and let them interact with each other subject to the standard rules for "Life." A competitive "Life" game, LIFE-2, has been devised which allows two players to put 3 life forms on the board per move and try to eliminate the opposing player. Class members may play this game as individuals or as teams. Are there any playing strategies which can be rigorously stated? Can two good players go on forever with no winner?

Computers are capable of . . . stretching human reason and intuition, much as telescopes or microscopes extend human vision. I suspect that the effects of this stretching will be as far-reaching as the effects of writing.

—Anthony G. Oettinger
(Professor, Harvard University)

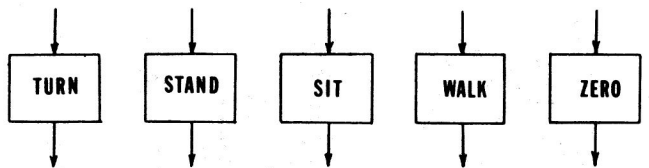
THE MECHANICAL MAN

The Mechanical Man, in the words of author Ken Lebeiko of Lockport (Illinois) High School, "is a fun way to learn flowcharting." Ken has had success with this creative presentation at all levels—from grade school to college students and suggests that related homework assignments are an ideal way to get the whole family involved in the learning process! The Mechanical Man is a good activity-oriented learning idea for incorporation into a computer familiarization course or an introduction to computer science and programming!

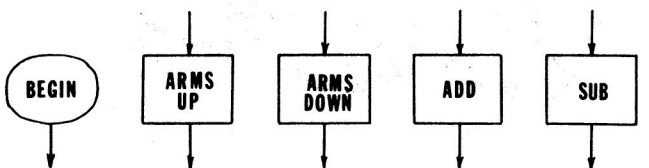
HANDOUT:

A mechanical man is sitting on a chair, facing a wall. Draw the flowchart of the procedure to walk the mechanical man to the wall and return him to his initial position.

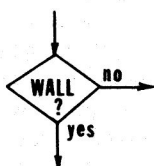
Permissible flowcharting symbols are:



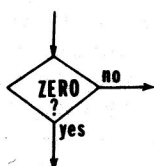
Turn 90° to the right Stand up Sit down Take one forward step Set the counter to zero



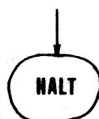
Start the procedure straight out Raise arms straight out Lower arms Add 1 to the counter Subtract 1 from the counter



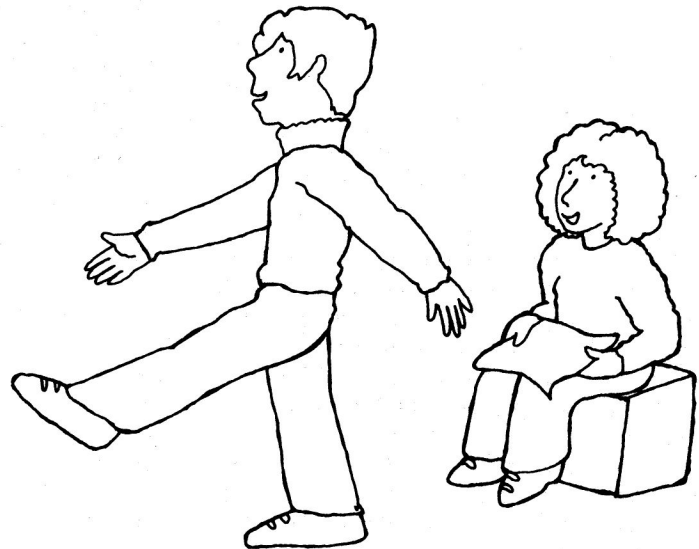
Test for touching wall (arms must be straight forward to activate the finger contact units)



Test if counter is zero



Stop the procedure



TEACHING INSTRUCTIONS:

1. Give students this handout and explain the problem. Demonstrate the procedure by example—put a chair six feet from the classroom wall, etc.
2. When students have completed the flowchart, choose several samples (with errors, of course!) to be placed on the blackboard. Slower students will gain from this exercise as each flowchart is explained.
3. Use example flowcharts without counters first. Later, demonstrate the use of counters in flowcharting and show the cumulative effect of not clearing a counter.
4. Once the initial explanation of the first flowchart is complete, the student-author becomes the mechanical man and "acts out" his flowchart. When errors are detected, DON'T change the flowchart. Instead, enjoy the fun of discovering how the error affects the final outcome. (Common occurrences include sitting on the floor, walking off in a random direction, etc.!)
5. While the activity and interest level is still high, separate students into groups to evaluate each others' flowcharts. Find the "perfect" flowchart, and put it on the blackboard.
6. Sit the member of the group with the "perfect" flowchart in a chair *against* the wall (leaving room for legs, of course!) Is the flowchart really "perfect"? Does it work for this special case? How can the flowchart be changed?
7. Now try the special case where the chair is one step from the wall and answer the same questions posed in Item #6.

KINGDOM

by Lee L. Schneider
University of Wisconsin
Milwaukee, Wisconsin

Todd L. Voros
Marquette University
Milwaukee, Wisconsin

The story behind the game of KINGDOM is an interesting one, not so much for the attributes of the game itself (although we consider them quite interesting, too), but because this game was eventually used for the purpose of teaching a computer how to play a game by itself. Although at first glance this might seem to be a silly thing for a computer to be doing, it should be pointed out that many of the discoveries in pattern recognition, decision making, and "self-optimizing" or learning programs began with some simple research into the playing of games.

The story of this particular game began rather late one night, as we were attempting to utilize the features of OS/8 running on a DIGITAL PDP-8/E system, owned by the Marquette University chemistry department. Having discovered the FOCAL interpreter of OS/8, we tried a few short test programs and began to search for something a bit longer to use for an extended test. We had all but given up when, in a somewhat-deserted drawer in a somewhat-unused corner of the lab, we came upon a paper tape for a game program called HAMURABI.

About two hours later, we reached the unanimous conclusion that this game had some definite possibilities; we had compiled a rather large collection of notes and ideas on changes, improvements, additions, and expansions which could be used as the basis of a new game. This new game, KINGDOM, would be based on HAMURABI but would have a much greater range of options and player responses.

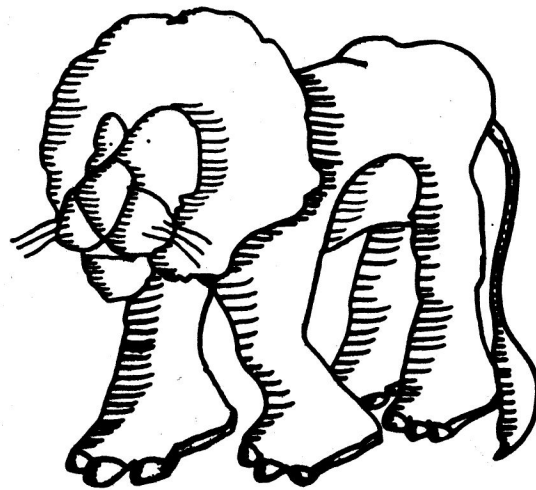
At that time, one of the authors was a student at the Milwaukee Technical College, as well as a part-time employee of the physics department there. Between the times spent writing data analysis programs for physics labs, a version of the new game was put together in FORTRAN (an obscure language known only to certain segments of the non-BASIC-speaking world). The end result, after two or three weeks of constant trial and revision, was a timesharing game program—KINGDOM!

THE PREMISE

The player is assumed to be ruler of a mythical medieval kingdom. The ruler is expected, at the beginning of each year of his reign, to make decisions which will maintain the kingdom and advance him toward the goal.

THE GOAL

The ultimate goal of a KINGDOM player is to rule the world. This rather megalomaniacal goal is obtained by attaining a total wealth of one billion units (the sum total of bushels plus acres owned). Naturally, the player must keep his kingdom maintained for a long enough time to do this; should all his population die out, disaster occurs and his empire crumbles. Needless to say, persons who are naturally inclined toward extreme greed make excellent KINGDOM players.



THE PLAY

At the beginning of each year, a report is printed giving the vital statistics for that year, including population and total wealth. The player must first decide on the amount of land to be bought or sold in exchange for grain. The player next selects the amount of grain to be distributed for food, and the number of acres to be planted that year. After the player has completed the question/answer sequence for the current year, control is relinquished to the KINGDOM program, and the player proceeds to reap the benefits and/or wrath of what he has wrought. In addition, as random events occur throughout the year, their effect on the player's KINGDOM is noted by appropriate printouts.

THE RULES

To keep the average greedy player from getting rich too quickly, a number of rules are in effect. These can be divided into two general categories, game constants and random factors, as follows:

GAME CONSTANTS

The following factors remain constant throughout the play of the game:

- 1) **LABOR**—Each subject of the kingdom is able to plant and harvest a maximum of two acres of land per year.
- 2) **SEED**—Each acre planted requires three bushels of grain for seed.
- 3) **FOOD**—Each subject requires a minimum of ten bushels per year for food.

RANDOM FACTORS

These factors occur at random intervals throughout the game: some have a fixed percentage of occurrence, others depend upon conditions present at time of testing. Numbers in parentheses indicate the probability of an event occurring in any given year. The random factors are:

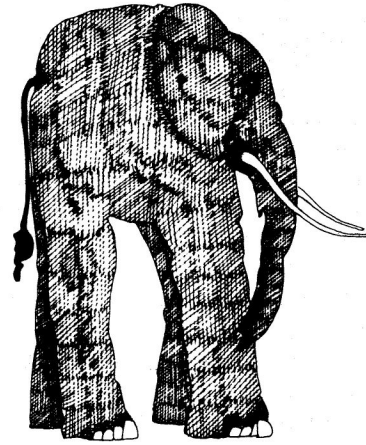
- 1) **HARVEST**: The normal variation of the harvest is between 3 and 6 bushels per acre. This range is in turn affected by two other factors:

RAIN: Increases the harvest to 7, 8 or 9 bushels per acre. (15%)

DROUGHT: Decreases the harvest to 1 or 2 bushels per acre. (15%)

The trading price of land varies according to the harvest; that is, if the harvest for a given year is 5 bushels per acre, the trading price for land will be 5 bushels per acre.

- 2) **LOSS TO RATS**: Due to the lack of efficient exterminators during medieval times, rats consume a portion of the harvest, which may range from none up to half of the yearly harvest.
- 3) **FOOD RIOTS**: This is a form of social protest which occurs when too many people are underfed.
- 4) **FOOD SURPLUS**: Besides a tremendous gain in the ruler's popularity, the surplus may cause an extra increase in population, on a rough scale of one new subject for each 20 bushels of excess food. (This is known as the Adam and Eve Law.)
- 5) **NORMAL POPULATION VARIATIONS**: Aside from the various outside influences, the population may be expected to vary each year due to new births and immigrations and death by natural causes.
- 6) **PLAGUE**: The occurrence of plague may wipe out from thirty to eighty percent of the population. (10%)
- 7) **THEFT**: Despite all precautions, an occasional thief makes off with up to a tenth of the grain from the storage house. (10%)
- 8) **EARTHQUAKE**: An earthquake may destroy from ten to fifteen percent of the land in the kingdom. (10%)



- 9) **ATTACK BY THE HUNS**: Having nothing else to do, a wandering band of itinerant Huns occasionally attacks the kingdom, stealing from five to ten percent of your grain, burning up to one-fiftieth of your land, and killing off twenty to forty percent of your population. (10%)

- 10) **GRAIN SHIPMENTS**: Not all random occurrences are unhealthy; a sudden appearance of Robin Hood with bushels of grain in hand has saved many kingdoms in peril.

- 11) **LAND RECLAIMING**: Land may be reclaimed from the ocean or desert. (15%)

Before long, most of the "bugs" were worked out of game, and most players agreed that the present form was of sufficient merit to provide both an interesting and a challenging problem to the player. But now the question arose; what was the best strategy for winning?

One interesting development was that players differed on the definition of a "good" technique; some said it was a strategem which produced a win *quickly*, while others said it should produce a win *consistently*. As it turned out, neither group could come up with a "foolproof" plan to achieve either goal.

AN OPTIMUM STRATEGY?

After extensive play at both the University of Wisconsin and Marquette University it appeared that no two players could agree exactly on the best technique. As we attempted to analyze the data, it became apparent that in order to "smooth out" the effect of random variables, we required data for large numbers of games, and a method of reducing this data to show general trends in the progress of the player.

This gave us three options from which to choose:

- 1) we could wait for the next fifty years, during which
- 2) we could find one or two players willing to sit down and play several thousand games using each possible strategy, 3) we could program another machine to play the game, reduce the data, and give us the results we wanted.

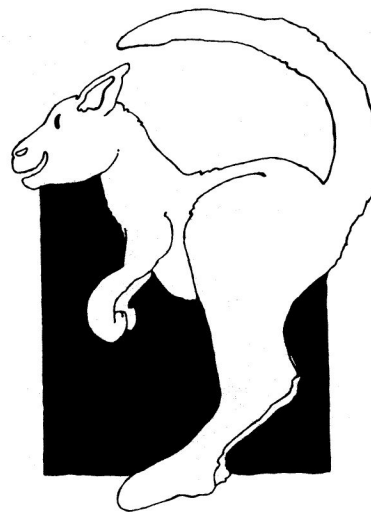
Thus the program RAMSES was born. RAMSES can play several thousand games of KINGDOM within ten or fifteen minutes, log the results, point out exceptional

Continued on p. 53

KANGAROO

Edward Lupin was a sophomore at Mount Pleasant High School in Wilmington, Delaware, when he wrote this program in BASIC-PLUS for the school's PDP-11 RSTS/E system. Mount Pleasant is one of eighteen secondary schools in the state-wide Project DELTA consortium.

The author's mathematics teacher, Mr. David Van Wickle, submitted the program to EDU in hopes that mathematics teachers and students who have grown "tired of the standard plotting exercises . . . for linear equations" would benefit. Edward's program will plot up to thirty linear equations on the same plane—and it even plots kangaroos. See for yourself, *mite!*



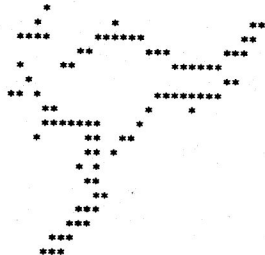
PROGRAM LISTING:

```
EDD13 08:55 AM 08-JAN-73
10 OPEN "ED.MAT" FOR OUTPUT AS FILE 1
20 INPUT "INSTRUCTIONS "JAS :IF AS<>"YES" THEN 50
30 OPEN "EDD13" FOR INPUT AS:PRINT AS:NEXT L
40 FOR L=1 TO 7:INPUT LINE #2,AS:PRINT AS:NEXT L
50 INPUT "X-AXIS HOW FAR FROM TOP "JAX
60 DIM #1,A(30),B(30),C(30),D(30),H(50),I(2)
70 MAT A=ZER:MAT B=ZER:MAT C=ZER:MAT D=ZER:MAT H=ZER
80 INPUT "HOW MANY EQUATIONS ARE THERE "JE
90 FOR F=1 TO JE
100 INPUT A(F),B(F),C(F),D(F)
110 NEXT F
120 DEF FNR(A)=INT(A+.5)
130 DEF FNY(A,B,X)=INT(-1*(A*X+B)+.5)+X
140 FOR F=1 TO JE
150 IF A(F)=0 THEN 190
160 FOR X=C(F) TO D(F)
170 H(FNY(A(F),B(F),X),FNR(X+30))=1
180 NEXT X:GO TO 220
190 FOR Y=D(F) TO D(F)
200 H(FNR(A(F)-Y),FNR(B(F)+30))=3
210 NEXT Y
220 NEXT F
230 S=0
240 PRINT:PRINT
250 FOR I=1 TO 50
260 IF S=1 THEN 310
270 FOR K=1 TO 72
280 IF H(I,K)<>0 THEN S=1
290 NEXT K
300 IF S=0 THEN 350
310 FOR J=1 TO 72
320 IF H(I,J)=0 THEN PRINT " "; ELSE PRINT "*";
330 NEXT J
340 PRINT
350 NEXT I
360 CLOSE 1:KILL "ED.MAT"
370 CLOSE 1,2
380 END
```

SAMPLE RUN:

```
EDD13 09:58 AM 09-JAN-73
INSTRUCTIONS ? YES
THIS PROGRAM GRAPHS UP TO THIRTY LINEAR EQUATIONS ON THE SAME PLANE
THE EQUATIONS ARE INPUTTED IN THIS MANNER :
1. FOR Y=3*X+5 , -2<=X<=5 DO THIS ? 2,0,3,5
2. FOR Y=2*X , -1<=X<=5 DO THIS ? 0,3,-1,6
3. FOR X=-1 , -4<=Y<=3 DO THIS ? 10E10,-1,-4,3 WHERE 10E10 IS A FLAG
   THAT TELLS THE COMPUTER IT IS A NO SLOPE EQUATION
X-AXIS HOW FAR FROM TOP ?
HOW MANY EQUATIONS ARE THERE ?
```

? 2,24.5,-11,-10



DOUBLE REVERSE

If you recall, the Program of the Month featured in EDU #10 was REVERSE—the BASIC computer game that challenges the player to resequence a list of numbers through a series of "reverses." Brother J. F. O'Connell from St. John's Preparatory School in Danvers, Mass., recently submitted another REVERSE—a version of the same game written in FOCAL. Most of the translation is credited to Jeff Berman, an 11th grade student. REVERSE in FOCAL is reproduced here for our many faithful FOCAL users. In addition, an "expert" rating has been awarded to James Feenan who beat the game in only 7 (yes!) moves.

C-FOCAL, 1969

```
01.10 T !!"THE GAME OF 'REVERSE'. TO WIN YOU HAVE
01.20 T !!"TO ARRANGE A LIST OF NOS. (1 THROUGH 9)
01.30 T !!"IN NUMERICAL ORDER FROM LEFT TO RIGHT. TO MOVE
01.40 T !!"YOU TELL ME HOW MANY NOS. (COUNTING FROM LEFT)
01.50 T !!"TO REVERSE. FOR EXAMPLE, IF THE LIST IS:
01.60 T !!"2 3 4 5 1 6 7 8 9
01.70 T !!"AND YOU REVERSE 4, THE RESULT WILL BE
01.80 T !!"5 4 3 2 1 6 7 8 9
01.85 T !!"NOW, IF YOU REVERSE 5, YOU WIN!
01.90 T !!"1 2 3 4 5 6 7 8 9
01.95 T !!"HOPE YOU LIKE THIS GAME! IF YOU WANT TO QUIT,
01.97 T " REVERSE 0"IG 2.1
```

```
02.01 E 1
02.10 S N=9:F K=1,N:D 2.3
02.20 G 2.35
02.30 S A(K)=FTR<N*(2*(FRAN( )-.5))>+1
02.35 F K=1,N:D 2.4
02.37 G 3.05
02.40 F J=1,K-1:I [A(K)-A(J)]2.5,2.3
02.50 C
```

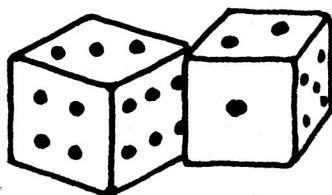
```
03.05 T !!"THE LIST IS:"JS T=0:D 6.1
03.10 A !!"HOW MANY SHALL I REVERSE?"R:I (R)3.2,3.7
03.20 I (R-N)3.4,3.4:T !!"I CAN REVERSE AT MOST"N:G 3.1
03.40 S T=T+1:F K=1,FTR(R/2):D 4
03.45 G 6.1
03.50 S S=0:F K=1,N:D 8
03.55 I (S-9)3.1,3.6
03.60 T !!"YOU WON IT IN"T," MOVES!"!
03.70 A !!"TRY AGAIN? (YES OR NO)"AN:I (AN=0YES)3.8,2.1
03.80 T !!"HOPE YOU HAD FUN!"!IG
```

```
04.10 S Z=A(K):S A(K)=A(R-K+1):S A(R-K+1)=Z
04.20 R
```

```
06.10 T !!"F K=1,N:T X2,A(K)
06.20 G 3.5
```

```
08.10 I (A(K)-K)8.3,8,2,8.3
08.20 S S=S+1
08.30 R
*
```


EDUCATIONAL GAME REVIEWS



Simulations and games are becoming a widely accepted and popular teaching technique. This is because they create an intense and enjoyable experience for the participant, foster a spirit of cooperation among players, and are an extremely effective way of teaching about complex interactions and systems.

Over the past months, the EDU editors have obtained a substantial number of manual as well as computerized educational games and simulations. Several manual games were reviewed in EDU #11; capsule reviews of additional games are printed here. Try adapting some to the computer!

LAND USE: Community needs and environmental protection.

Participants plan the use of a 10-acre site of land. The community needs more homes and roads, but the people want to preserve as much of the natural environment (clean water, trees, wildlife, habitat, grass, etc.) as possible. Players learn concepts of cluster zoning, planning units and community development.
(Up to 6 players; \$1.95; Education Ventures)

NEW HIGHWAY: Resolve highway building issues. Players assume roles of public officials, small businessmen, manufacturers, and community residents and lobby for their concerns about an unsafe and congested highway in a public hearing. Votes are tallied at the end of each round of play.
(6 players; \$4.95; Education Ventures)

BUDGETS & TAXES: Resolve a town's annual budget. Players assume roles of voters and public officials in different town departments and influence the annual budget for their town. Votes are taken at the end of each round of play.
(6 players; \$4.95; Education Ventures)

THE CITIES GAME: Simulates the realities of city life. This is a role-playing board game of negotiation, political bargaining, and coalitions. Players face crises (excess garbage, delayed construction, higher taxes, etc.) and must cooperate and compromise to find solutions. The alternatives, if solutions aren't found, are riots, police action, and unpredictable change.
(4 to 12 players; Dynamic Design)

SOURCES:

Dynamic Design Industries	Education Ventures
1433 N. Central Park	209 Court Street
Anaheim, CA 92802	Middletown, CT 06457

KINGDOM

Continued from p. 51

occurrences and general trends in performance, and even modify its own strategy to improve its play, the last of which is still under investigation and development.

Teaching the machine to play the game is not as easy as it first appears. Not only must the second segment (PLAYER program) be responsive to the parameters we supply, but it must also play legally, within the rules of the game, *regardless of the parameters it is given to play with*. Once satisfied that PLAYER performed correctly and within the rules, the next step was to put the PLAYER and KINGDOM programs in contact with each other.

This third and final link of the RAMSES program, called MASTER, is still undergoing study. In its present state, MASTER can play the game in blocks of 100 games apiece, each under the same parameters. This allows us to "average out" random fluctuations in any particular set of game conditions. Each block of 100 games is called a Dynasty; ten Dynasties make up an Era; for each Dynasty, parameters are kept at a fixed level, and the one hundred games are analyzed for best, worst, and average values that relate to performance of the algorithm currently in use. By comparing the results encountered over an Era (each Dynasty played with one parameter slightly changed) we are able to watch the effect of that parameter on the ability of the machine to play the game well. We can change the parameters selectively, or allow the machine to change them itself either in sequence or at random.

As with human players, the machine is able to analyze a strategy from one of two viewpoints: stability of performance or ability to attain the goal within a given amount of time. As with human players, the machine has found that certain parameter values produce very long reigns (some in excess of 150 years), but do not win except by accident. Others produce occasional spectacular results, but fail miserably the rest of the time. The latter we call the "Long Shot" approach, allowing the machine to "go for broke" whenever the opportunity presents itself, with little regard for the future safety of the game. Such strategems have produced games which were won in less than fifty years (which is usually considered masterful play for a human player); as of this time, the all-time speed record for the machine is one single 34-year win. The all-time record for a human player is 43 years, set by a Marquette Computing Center staff member.

We cannot say that it is impossible to find a strategy which will win 100% of the time; however, because of the construction of the game, we can say that no win will ever be possible in less than 16 years.

We would be glad to supply the source listings of KINGDOM in FORTRAN, BASIC or FOCAL to anyone who sends us a stamped, self-addressed envelope. We also welcome any suggestions or comments you may have to make on our project or the game itself.

PROGRAM OF THE MONTH: LIFE



Computer scientists and aspiring computer experts alike have always had an unusual fascination for John Conway's game of "Life." This version, LIFE, was written by *Clark Baker, Project DELTA, Newark, Delaware*, and was originally published in DIGITAL's 101 BASIC Computer Games book. (See article in this issue entitled "The Game of Life.")

PROGRAM LISTING

```

1 REM LIFE CLARK BAKER 3/72 C.O.O.
2 PRINT CHR$(31);CHR$(29);CHR$(31);"ENTER YOUR PATTERN:"
3 X1=1:Y1=1: X2=24:Y2=78
10 DIM A$(24,78),B$(24)
20 OPEN "KB;" AS FILE 1
30 ON ERROR GO TO 80
40 CX=1
50 INPUT LINE #1,B$(CX):B$(CX)=LEFT$(B$(CX),LEN(B$(CX))-2)
60 CX=CX+1
70 GO TO 60
80 LX=0
90 FOR XX=1 TO CX-1
100 IF LEN(B$(XX))>LX THEN LX=LEN(B$(XX))
110 NEXT XX
120 X1=X1+CX/2
130 Y1=Y1+LX/2
140 FOR YX=1 TO CX
150 FOR YX=1 TO LEN(B$(XX))
160 IF MID$(B$(XX),YX,1)<>" " THEN A$(X1+XX,Y1+YX)=1:PX=PX+1
170 NEXT YX
180 NEXT XX
200 PRINT CHR$(29);CHR$(30);
210 PRINT "GENERATION:";GX;"POPULATION:";PX;CHR$(30);IF I% THEN PRINT "INVALID!"
215 X3=24:Y3=78:X4=X4+1:Y4=Y4+1:PX=0
220 GX=GX+1
225 PRINT CHR$(13);CHR$(10);CHR$(30); FOR XX=1 TO X1-1
230 FOR YX=Y1 TO Y2
240 PRINT
250 FOR YX=Y1 TO Y2
253 IF A$(XX,YX)=2 THEN A$(XX,YX)=0:GO TO 270
256 IF A$(XX,YX)=3 THEN A$(XX,YX)=1:GO TO 261
260 IF A$(XX,YX)<>1 THEN 270
261 PRINT TAB(YX);" "
262 IF X3<X4 THEN X3=X4
264 IF X3>X4 THEN X4=X3
266 IF Y3<Y4 THEN Y3=Y4
268 IF Y3>Y4 THEN Y4=Y3
270 NEXT YX
280 PRINT CHR$(30);
290 NEXT XX
295 PRINT CHR$(30); FOR XX=X2+1 TO 24
298 PRINT CHR$(29);
299 X1=X3:X2=X4:Y1=Y3:Y2=Y4
301 IF X1<X2 THEN X1=X2:Y1=Y3:Y2=Y4
303 IF X2>X2 THEN X2=X1:Y1=Y3:Y2=Y4
305 IF Y1<Y2 THEN Y1=X3:Y1=Y4
307 IF Y2>Y2 THEN Y2=X4:Y1=Y4
309 PX=0
500 FOR X1=X1-1 TO X2+1
510 FOR Y1=Y1-1 TO Y2+1
520 CX=0
530 FOR IX=X1-1 TO X1+1
540 FOR JX=Y1-1 TO Y1+1
550 IF A$(IX,JX)=1 OR A$(IX,JX)=2 THEN CX=CX+1
560 NEXT JX
570 NEXT IX
580 IF A$(X1,Y1)=0 THEN 610
590 IF CX<3 OR CX>4 THEN A$(X1,Y1)=2 ELSE PX=PX+1
600 GO TO 620
610 IF CX=3 THEN A$(X1,Y1)=3:PX=PX+1
620 NEXT YX
630 NEXT XX
635 X1=X1-1:Y1=Y1-1:X2=X2+1:Y2=Y2+1
640 GO TO 210
650 END

```

GENERATION: 1

POPULATION: 6

GENERATION: 21

POPULATION: 27

GENERATION: 22

POPULATION: 37

SAMPLE RUN

ENTER YOUR PATTERN:

```

**
**

```

~2
GENERATION: 0

POPULATION: 5

```

*
**
**
**
**
**
**
**
**
**

```


FREE!! EDU SUBSCRIPTION FORM

This form *must* be returned to receive EDU in the 1974-75 school year. Please be sure to fill in all the information requested on both sides of the form.

WHAT DO YOU THINK OF EDU?

1. What do you think of the following sections of EDU?

More, more!	Just right	Of no Interest	
_____	_____	_____	Info on DIGITAL products
_____	_____	_____	Info on non-DIGITAL products
_____	_____	_____	User application stories
_____	_____	_____	Articles and papers
_____	_____	_____	Conference and meeting schedules and reports
_____	_____	_____	Source materials—Computer
_____	_____	_____	Source materials—Non-computer
_____	_____	_____	Problems and games
_____	_____	_____	Classroom ideas
_____	_____	_____	Book reviews
_____	_____	_____	History and operation of DIGITAL
_____	_____	_____	Opinions and editorials
_____	_____	_____	Letters to the Editor

2. Are you willing to contribute articles, ideas, or stories to EDU?

☐ Yes, I'd love to.

☐ No, I don't have the time.

☐ _____

3. I heard about EDU from:

☐ Educational Periodical

☐ Colleague

☐ Trade Show

☐ DIGITAL Personnel

☐ Other _____

4. How do you feel we could improve EDU? _____

5. Current or planned computer usage

☐ DIGITAL User—Model _____

☐ Other User—Manufacturer and Model _____

☐ Timesharing Service or RJE User

☐ No Computer—Plan to get one

When? ☐ Next 12 Mos. ☐ 1 to 2 years ☐ Longer

☐ No Computer—No plans to get one

6. Type of Institution

☐ Secondary School—Dept.? ☐ Math ☐ Science ☐ Other _____

☐ Vocational/Technical School

☐ Community or Junior College

☐ College or University Computer Center

☐ College or University Department

Which Dept.? _____

7. Application Area(s) of Interest

☐ Instructional ☐ Administrative ☐ Research ☐ Other _____

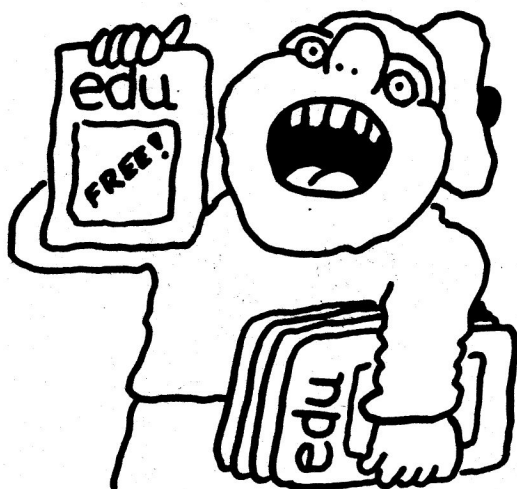
BE SURE TO FILL IN NAME AND ADDRESS INFORMATION ON THE OTHER SIDE OF THIS FORM

First Class
Permit #33
Maynard, Mass.

Business Reply Mail. No postage necessary if mailed in United States

Postage will be paid by:

Education Products Group
Digital Equipment Corporation
Maynard, Mass. 01754



Fill in name and address if label below is incorrect or not attached.

Name _____

Title _____

Dept. _____

School _____

State _____ Zip _____

digital

edu

Digital Equipment Corp.
Maynard, Mass. 01754

BULK RATE
U.S. POSTAGE
PAID
PERMIT #15
Maynard, Mass.

ATTN: Director of Educational Computing